2011-06

Assumptions, trust, and names in computer security protocols

Shearer, Charles Dylan

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ASSUMPTIONS, TRUST, AND NAMES IN COMPUTER SECURITY PROTOCOLS

by

Charles Dylan Shearer

June 2011

Thesis Advisor:	George W. Dinolt
Second Reader:	James Bret Michael

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–6–2011	Master's Thesis	27-09-2010—17-06-2011

4. TITLE AND SUBTITLE

Assumptions, Trust, and Names in Computer Security Protocols

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Charles Dylan Shearer

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

N/A

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

13. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

14. ABSTRACT

A major goal of using any security protocol is to create certain beliefs in the participants. A security protocol will use techniques like cryptography to guarantee some things, but it will still require a participant to make assumptions about other things that the protocol cannot guarantee; such assumptions often constitute trust in other participants. In this thesis, we attempt to precisely identify the required assumptions of some example protocols. In the process, we find that we must consider the names that participants use to reason about each other. It turns out that naming is a complex topic with a rich body of philosophical work, and we apply some ideas from this work to the problem of identifying security protocols' required assumptions. Finally, we begin work on a mathematical model of protocols and beliefs to which a formal logic of belief could be applied. The model is left incomplete because of some unresolved problems with modeling belief caused by the design requirement that the model's elements have clear operational meanings. The solution of these problems is left as future work.

15. SUBJECT TERMS

computer security, protocol, assumption, belief, trust, naming

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	UU	85	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

ASSUMPTIONS, TRUST, AND NAMES IN COMPUTER SECURITY PROTOCOLS

Charles Dylan Shearer
Civilian
B.S., Kent State University, 2007
B.A., Kent State University, 2007

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 2011

Author: Charles Dylan Shearer

Approved by: George W. Dinolt
Thesis Advisor

James Bret Michael
Second Reader

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

A major goal of using any security protocol is to create certain beliefs in the participants. A security protocol will use techniques like cryptography to guarantee some things, but it will still require a participant to make assumptions about other things that the protocol cannot guarantee; such assumptions often constitute trust in other participants. In this thesis, we attempt to precisely identify the required assumptions of some example protocols. In the process, we find that we must consider the names that participants use to reason about each other. It turns out that naming is a complex topic with a rich body of philosophical work, and we apply some ideas from this work to the problem of identifying security protocols' required assumptions. Finally, we begin work on a mathematical model of protocols and beliefs to which a formal logic of belief could be applied. The model is left incomplete because of some unresolved problems with modeling belief caused by the design requirement that the model's elements have clear operational meanings. The solution of these problems is left as future work.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

Acknowledgements

I thank my advisor, Prof. George Dinolt, and my second reader, Prof. Bret Michael, for their advice and encouragement while I was working on this thesis. I thank Dr. Paul Bohan Broderick (whose seminar, Philosophy & Cognitive Science, is the best class I have taken) for teaching me about formal logic and the philosophy of computer science. I thank Dr. Michael Collard, who was my advisor for my undergraduate thesis, for showing me how to do research. I am able to attend NPS because of DoD's SMART Scholarship, and so I thank the U.S. Congress for establishing this scholarship, the SMART Scholarship Program Office for awarding it to me, and U.S. Army CERDEC for sponsoring me, as well as the U.S. taxpayers for funding it. Finally, I thank my cat Uzu for her support while I was working on this thesis — in fact, several chapters are the result of her walking on my keyboard.

THIS PAGE INTENTIONALLY LEFT BLANK

There are four sorts of men:
He who $\neg K$ and $\neg K \neg K$: he is a fool — shun him;
He who $\neg K$ and $K \neg K$: he is simple — teach him;
He who K and $\neg K K$: he is asleep — wake him;
He who K and $K K$: he is wise — follow him.

Paraphrased Arabian proverb.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

1.1 Beliefs in Security Protocols

In April 2011, the White House released "National Strategy for Trusted Identities in Cyberspace" (NSTIC) ([1]), which describes a plan for encouraging the widespread use of a particular style of online identification and authentication often called 'federated identity', 'federated authentication', and 'federated identity management' (e.g., [2] and [3]). If the plan is successful, Americans will log into their various online accounts — email accounts, bank accounts, Twitter and Facebook and Skype accounts, etc.— using identifiers and credentials issued, or at least vouched for, by a third-party "identity provider," instead of using a potentially different username-password pair for each account. With this scheme, the White House hopes to address problems such as the burden on users of maintaining multiple username-password pairs and the burden on service providers of performing authentication for each account; moreover, they believe that this scheme will provide authentication mechanisms with even greater assurance than is currently possible, so that services currently deemed too sensitive to offer online will become available.

The public draft of this plan, released in June 2010, was the primary motivator for this thesis. This thesis is, in part, about the difference between "traditional" mechanisms of online identification and authentication and the kind of mechanisms advocated in NSTIC. This difference is not in the strength of encryption, the quality of the credentials, or the sensitivity of the accounts and activities that are protected, and yet the result is that the White House's mechanisms are riskier than traditional ones.

The difference is that NSTIC's mechanisms require more *trust* on the part of participants. Trust in this context can be thought of as a belief about a person or entity that one must have in order to act and whose truth is important to one's well-being. This thesis presents a view of security protocols in general that emphasizes the beliefs that participants are supposed to have (the "required assumptions") before using the protocol, in order for it to achieve its goals. This view also presents security protocols' goals as beliefs (the "goal beliefs") that the protocols are supposed to induce in the participants. In other words, a security protocol presents an argument to a participant, based on the participant's assumptions, whose conclusions are the goals (for

that participant) of the protocol. One of our observations will be that, the more trust that a protocol requires, the riskier it is (all else equal).

Another of our observations will be that identifying a protocol's required assumptions, and the beliefs that it is supposed to induce, is not trivial. Trust is just one kind of assumption that participants may be required to make — some other kinds are assumptions about the meanings of messages sent during a run and assumptions about the beliefs of other participants. Additionally, we will see that, to describe these beliefs, we must account for the *names* that participants use to refer to people, other participants, or themselves. Indeed, naming is an overlooked but very important aspect of the use of security protocols.

1.2 Structure of the Thesis

The main parts of this thesis are Chapter 4 and Chapter 5. In Chapter 4, we continue the discussion begun in the previous section by formulating appropriate goal beliefs and required assumptions for five hypothetical applications of security protocols. In the process, we find that these beliefs and assumptions can be quite complex and subtle, even for simple protocols, and require us to answer some difficult philosophical questions. In Chapter 5, we take a first step toward a rigorous model of protocol applications that will enable us to use a formal language to specify goal beliefs and required assumptions and even a formal logic to show how the assumptions imply the goal beliefs. We encounter a problem with applying the possible-worlds model of belief, and, after describing it, we leave it as future work.

Chapter 2 reviews modeling techniques and formalisms that are important to any attempt at building a rigorous model of protocol applications and belief — namely, multi-agent systems and modal logic. Chapter 3 presents important works from computer security, formal logic, and philosophy that are relevant to this thesis, and reviews several of them in more detail.

Finally, Chapter 6 presents conclusions and areas of useful future work.

This thesis includes some mathematical content that builds on the discussion of beliefs in security protocols. It is important, and a reader who is interested in applying formal methods to security protocols should read this thesis straight through. But the mathematical content is not essential; the main ideas are not about mathematical models but about security protocols, belief, naming, and meaning. A reader who is not interested in formal methods should read the following parts, in sequence:

1. The rest of the Introduction.

2. Section 3.3

3. Section 3.5

4. Chapter 4

5. Chapter 6

1.3 Attributing Belief to Participants

Traditionally, the term 'principal' is used to denote the things that perform roles in protocols, and researchers have been comfortable attributing not only actions but also beliefs to these things. This leads to difficult philosophical questions. Consider several ways of performing a protocol role. A computer may be set up to perform a role completely without human intervention (e.g., DNS servers). Or a computer may do most of the work but still interact with a person at some point, perhaps at the beginning and end of a run (e.g., the "client" role of HTTP) or somewhere in the middle (e.g., the "client" role of TLS when the user must decide whether to trust the server's certificate). Or a role may be performed entirely by a person (e.g., when someone uses a Telnet program to send an HTTP request). If 'principal' denotes just the entity performing the actions, it is hard to specify the meanings of statements like 'Principal A believes X' without explaining how a computer can have beliefs when no human is involved in performing a role. At the same time, if 'principal' refers just to the person or organization responsible for the computer that performs the actions, then we cannot attribute those actions to the principal.

It thus seems that the properties we attribute to a principal — actions, knowledge, beliefs — are often shared between one or more computers and one or more people who can access and control those computers. I will use 'principal' to denote a combination of all entities that together perform a protocol role. If a role is performed by a computer and you are not comfortable attributing belief to computers, you can take sentences like 'Principal A believes X' to mean that the person or people who can access and control the computer *would* believe X if they were to review the data currently in the computer.

1.4 Notation

1.4.1 Quotation

Much of this thesis is about names and identifiers, and one of its points is that, to understand security protocols, we should be precise with regard to naming in our descriptions of principals' beliefs. In his paper on quotation, Donald Davidson wrote the following:

> When I was initiated into the mysteries of logic and semantics, quotation was usually introduced as a somewhat shady device, and the introduction was accompanied by a stern sermon on the sin of confusing the use and mention of expressions. [4, p. 27]

Judging from my experience and the books on logic that I have read, he was the only member of the congregation. (Quine was definitely present, but it might have been as the preacher — cf. [5, §4].) Nevertheless, at some points I would like to rigorously observe the distinction between using and mentioning expressions, in order to achieve that helpful precision in discussions of security protocols. To this end, I will use single quotation marks always to mention an expression and never in the ambiguous manner in which quotation marks are often used — in which an expression is both mentioned and used at the same time. However, this ambiguous use of quotation marks often has rhetorical value, and I will use double quotation marks ("scare quotes") for this purpose.

There will be places where I need to mention a class of expressions sharing a common form — for example, the set of expressions that consist of '$\mathbb{B}$' followed by a formula of some formal language. For this purpose, I will borrow Quine's idea and notation of *quasiquotation* ([5, §6]). For example, suppose we let 'ϕ' stand for an arbitrary formula of some formal language: with the quasiquotation marks '$\ulcorner$' and '$\urcorner$' we can denote the set of expressions that consist of '$\mathbb{B}$' followed by ϕ with the expression '$\ulcorner \mathbb{B}\, \phi \urcorner$'. For a quasiquotation to work, it must be clear which parts of a quasiquotation denote *themselves* (or, more precisely, the types of expression of which they are a token) (e.g., '$\mathbb{B}$') and which parts are *metavariables* — that is, stand-ins for other expressions (e.g., 'ϕ'). Throughout this thesis, I will use Greek letters like 'ϕ', 'ψ', and 'τ' as metavariables, since they do not occur in any formal language that I discuss and stand out among the symbols of these formal languages.

1.4.2 Protocol Messages

In Chapter 4 and occasionally elsewhere I need to describe the structures of protocol messages, which are often the result of cryptographic operations. I will use the following notation:

$$\ulcorner \alpha_1 \alpha_2 \urcorner \stackrel{\text{def}}{=} \ulcorner \text{the result of concatenating } \alpha_1 \text{ and } \alpha_2 \urcorner$$

$$\ulcorner \{\!|\alpha|\!\}_\kappa \urcorner \stackrel{\text{def}}{=} \ulcorner \text{the result of encrypting } \alpha \text{ with key } \kappa \urcorner$$

$$\ulcorner \kappa^{-1} \urcorner \stackrel{\text{def}}{=} \ulcorner \text{the inverse of key } \kappa \urcorner$$

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Background

2.1 Multi-Agent Systems

The mathematical model of security protocols that we begin in Chapter 5 is based on the multi-agent system framework, as presented in Fagin *et al.*'s book "Reasoning About Knowledge" ([6]). Using this framework, we start with a set $\mathcal{A} = \{a_1, \ldots, a_n\}$ of *agents*. At any particular time, each agent a_i is in some *local state* s_{a_i} that determines the properties of a_i that we are interested in (e.g., behavior, knowledge, beliefs). For each agent a_i, there is a set L_{a_i} of all possible local states. We may also want to model properties of non-agent objects (e.g., message-queues): we therefore include an *environment* e (along with its set L_e of possible local states) that also is in some local state s_e at any particular time. The whole system can be thought of as being in a *global state* that is the composition $(s_e, s_{a_1}, \ldots, s_{a_n})$ of the local states of the environment and all the agents at that time.

The system's state changes when an event that we are interested in occurs. Time is discrete, and we identify it with the natural numbers. (Time does not have to be discrete.) A *run* r is a function from time (i.e., $\mathbb{N}$) to the set of all possible global states (i.e., $L_e \times L_{a_1} \times \cdots \times L_{a_n}$) and thus is one possible history for the system. We identify our *multi-agent system* with a particular set $\mathcal{R}$ of runs. $\mathcal{R}$ reflects all the constraints and possibilities of the agents' actions. For any run r and time t, we call the pair (r, t) 'a point'. We use '$r(t)$' to denote the system's global state at point (r, t), and we use '$r_{a_i}(t)$' to denote the local state of agent a_i at point (r, t).

For example, if S is a system containing agents a and b that pass messages to each other, then there could be a run r such that $r_a(0)$ is a state in which a has a message M to send to b (but has not yet sent it), and $r_a(1)$ is a state that means that a has sent M to b; of course, $r_b(1)$ could mean that b has received M. If this is all we want to happen in r, then $r(1) = r(t)$ for $t > 1$ — that is, the system halts after time 1.

2.2 Modeling Belief with Modal Logic

Modal logic is a type of logic that deals with what are called 'modes of truth'. 'Modes of truth' is difficult to define, so I will just give some examples of sentences that modal logic is

particularly appropriate for formalizing:

9 is necessarily greater than 5.

One should not speed.

Bob thinks that the store is closed.

The modes of truth in these examples are necessity, obligation, and belief (respectively). In the meanings of each of these sentences, there is a proposition and a mode that is applied to that proposition specifying the way in which it is true. These two components are most evident in the last sentence. We can bring them out in the first two by rewriting them thus:

It is necessary that 9 is greater than 5. (2.1)

It is required that one does not speed.

In modal logic, we take a standard logic like propositional logic or first-order logic and add new operators (called 'modal operators') for specifying modes. For example, if we use '$\Box$' as a modal operator indicating necessity, and we use 'p' to denote the proposition that 9 is greater than 5, then we could formalize (2.1) thus:

$$\Box p$$

If we were using a quantified modal logic, we might formalize (2.1) instead thus:

$$\Box\,\mathrm{GreaterThan}(9, 5)$$

Propositional modal logic's standard semantics — possible-worlds semantics — involves "Kripke structures," named for Saul Kripke who helped create possible-worlds semantics (e.g., [7]). A model for non-modal propositional logic is an assignment of truth-values (i.e., $\{\mathbf{true}, \mathbf{false}\}$) to all proposition symbols (such a model is also called an "interpretation"); given such an assignment, the truth-value of any formula in propositional logic can be computed by applying the appropriate truth-functions (according to the operators in the formula) to the truth-values of its proposition symbols. While an interpretation of a propositional language represents the basic facts of the world, in possible-worlds semantics an interpretation represents

only one of the many possible worlds. The intuition for possible-worlds semantics is clearer if we consider necessity: some propositions may be true in the real world, but we can conceive of a world in which they are not (e.g., 'There are nine planets'); in contrast, other propositions are true in the real world, and it is not possible to conceive of a world in which they are false (e.g., '9 is greater than 5'). Accordingly, in our Kripke structure there are worlds in which there are nine planets and worlds in which there are some other amount of planets, but in all worlds 9 is greater than 5. Let 'p' stand for 'There are nine planets' and 'q' stand for '9 is greater than 5'. The modal formula '$\Box q$' asserts the fact that 9 is greater than 5 in all possible worlds.

Now consider belief. People have different beliefs, and this difference means that people disagree about which of the possible worlds is the *actual* world. If Bob believes that the store is closed, then for all worlds that Bob thinks might be the actual one, the store is closed. Of course, Bob's beliefs change, and thus the worlds that he considers possible change. We model this by putting Bob in one world w_1 in which he believes that the store is closed, and in another world w_2 in which he believes that it is open. Formally,

$$w_1 \vDash \text{`}\,\mathbb{B}\,s\text{'}$$
$$w_2 \vDash \text{`}\,\mathbb{B}\,\neg s\text{'}$$

('s' stands for 'The store is closed.' and '$\mathbb{B}$' is our modal operator for Bob's beliefs.) This is possible because the worlds that Bob believes are possible in w_1 and w_2 differ.

In general, for a set P of proposition symbols, a Kripke structure is a tuple $M = (\mathcal{W}, R_1, \ldots, R_n, \pi)$ where $\mathcal{W}$ is the set of all possible worlds, each R_i is a binary relation over $\mathcal{W}$, and π is an assignment of interpretations over P to worlds in $\mathcal{W}$. Thus, for any ϕ in P and any w in $\mathcal{W}$, $(M, w) \vDash \phi$ if and only if $\pi(w)(\phi) = \mathbf{true}$. For the truth-functional operators, satisfaction takes the standard form:

$(M, w) \vDash \ulcorner\neg\phi\urcorner$ if and only if $\pi(w)(\phi) = \mathbf{false}$
$(M, w) \vDash \ulcorner(\phi \wedge \psi)\urcorner$ if and only if $\pi(w)(\phi) = \mathbf{true}$ and $\pi(w)(\psi) = \mathbf{true}$
etc.

(Of course, not all modal languages have the same operators or use these symbols.) The lan-

guage must have exactly n modal operators — $\Xi_1, \ldots, \Xi_n$ — such that:

$$(M, w) \models \ulcorner \Xi_i \phi \urcorner \text{ if and only if } (M, w') \models \phi \text{ for every } w' \text{ such that } (w, w') \in R_i$$

(Cf. [6] and [8, Chapter 5] for more details on modal logic.)

2.2.1 Modeling Knowledge in Multi-agent Systems

It is straightforward to apply modal logic to a multi-agent system in order to model the knowledge of the agents in the system. [6] is devoted to showing how to do this and how to apply it to various problems. The main idea is that an agent's knowledge is determined by its local state, and therefore the agent will not be able to distinguish between any two points in which it is in the same local state. For a given multi-agent system $\mathcal{R}$ over a set $\mathcal{G}$ of global states and a set $\mathcal{A}$ of n agents, we describe the agents' knowledge by building a Kripke structure $(\mathcal{W}, K_1, \ldots, K_n, \pi)$. The set of possible worlds, $\mathcal{W}$, is the set of points. For any point (r, t), $\pi((r, t))$ is an interpretation of some set of proposition symbols that describes some facts about the global state $r(t)$ that we are interested in. For each agent i, K_i is a binary relation over points such that $((r, t), (r', t')) \in K_i$ if and only if i is in the same local state at (r, t) and (r', t').

In this thesis, we are interested in belief rather than knowledge, but the method of [6] is the starting-point from which (in Chapter 5) we begin developing a mathematical model of beliefs in security protocols.

2.3 Quantified Modal Logic of Belief

The modal logic presented above is a *propositional* modal logic; if we remove the modal operator, we are left with simple propositional logic. Propositional modal logic, in all its varieties, is well understood, in that there is a variety of classes of models from which to choose to represent one's particular modality of interest. This is not the case for quantified modal logic.

Quantified modal logic is a big topic, and I will not provide a comprehensive overview of it here. Instead, I will discuss two issues that are particularly important for this thesis: the interpretation of identifiers in quantified-modal-logic formulas, and the relation between quantifiers, modal operators, and the domain of quantification. Since we are only concerned with modal logics of belief, I will ignore other modalities like knowledge and necessity.

Languages for quantified modal logics are usually made by combining the syntax of proposi-

tional modal logics with the syntax of first-order non-modal logic. Models for quantified modal logics of belief usually take the form of a *relational* Kripke structure $M = (\mathcal{W}, B_1, \ldots, B_n, \pi)$, which is different from the Kripke structures for propositional modal logic in that π maps $\mathcal{W}$ to what are sometimes called 'relational structures'. These relational structures are simply the kind of thing that can serve as a model for a first-order language — that is, they have a special set that serves as the domain of the quantifiers, and they map constants to domain elements, function-symbols to functions, and predicate-symbols to predicates. Thus, for both propositional modal logics and quantified modal logics, π maps $\mathcal{W}$ onto models appropriate for the language.

The syntax and semantics of quantified modal languages vary, but the following is very common. For any w in $\mathcal{W}$, let '$w.Dom$' denote the domain of the quantifiers. For any w in $\mathcal{W}$, $\pi(w)$ assigns a member of $w.Dom$ to each constant; it assigns a function in w to each function-symbol; and it assigns a predicate in w to each predicate-symbol. Satisfaction ('$\models$') is defined defined over tuples of the form (M, w, V) where M is a relational Kripke structure, w is a possible world, and V is a "valuation" — that is, a partial function mapping variables to domain elements. But before defining satisfaction, we need to show how terms — i.e., constants, variables, and function-applications — are interpreted for a given (M, w, V). We will use the notation '$|\cdot|^{M,w,V}$' to denote the interpretation of a term relative to some (M, w, V). For any (M, w, V) (where $M = (\mathcal{W}, B_1, \ldots, B_n, \pi)$), we have the following:

$$
\begin{aligned}
|\kappa|^{M,w,V} &= \pi(w)(\kappa) && (\kappa \text{ is a constant}) \\
|\alpha|^{M,w,V} &= V(\alpha) && (\alpha \text{ is a variable}) \\
|\rho|^{M,w,V} &= \pi(w)(\rho) && (\rho \text{ is a function-symbol}) \\
|\Phi|^{M,w,V} &= \pi(w)(\Phi) && (\Phi \text{ is a predicate-symbol}) \\
|\ulcorner \rho(\tau_1, \ldots, \tau_m) \urcorner|^{M,w,V} &= |\rho|^{M,w,V}(|\tau_1|^{M,w,V}, \ldots, |\tau_m|^{M,w,V}) && (\text{each } \tau_i \text{ is a term})
\end{aligned}
$$

Satisfaction ('$\vDash$') is also defined recursively:

$(M, w, V) \vDash \ulcorner \Phi(\tau) \urcorner$ if and only if $|\tau|^{M,w,V} \in \pi(w)(\Phi)$

$(M, w, V) \vDash \ulcorner \forall \alpha \, \phi \urcorner$ if and only if $(M, w, V') \vDash \phi$ for any x in $w.Dom$
where V' is the same as V except for mapping α to x

$(M, w, V) \vDash \ulcorner \exists \alpha \, \phi \urcorner$ if and only if $(M, w, V') \vDash \phi$ for some x in $w.Dom$
where V' is the same as V except for mapping α to x

$(M, w, V) \vDash \ulcorner \Xi_i \phi \urcorner$ if and only if $(M, w', V) \vDash \phi$ for any w' in $\mathcal{W}$ such that $(w, w') \in B_i$
(Ξ_i is the modal operator corresponding to B_i)

$(M, w, V) \vDash \ulcorner \neg \phi \urcorner$ if and only if $(M, w, V) \nvDash \phi$

$(M, w, V) \vDash \ulcorner (\phi \wedge \psi) \urcorner$ if and only if $(M, w, V) \vDash \phi$ and $(M, w, V) \vDash \psi$

And so on for the other truth-functional operators.

2.3.1 Dealing with Names

Designing a model for a quantified modal logic of belief beyond what we have done in the previous paragraph involves some tricky questions about the nature of belief and the class of linguistic objects that includes names, descriptions, identifiers, etc.. For convenience, I will call such objects 'names'. Consider the use of names inside the 'that'-clauses in these sentences:

Alice believes that Obama is tall. (2.2)

Alice believes that the U.S. president is tall. (2.3)

If Alice for some reason (perhaps some sociological theorem) believes that the U.S. always elects tall presidents, then (2.3) could be true even if she does not know who the president is. If we were to model (2.3) with a relational Kripke structure, then in all the worlds that Alice considers possible, the U.S. president is tall but not necessarily the same person. In this way, the name 'the U.S. president' seems to refer to something abstract, perhaps the *role* of president, although in any particular world it certainly refers to exactly one person (I hope). In contrast, in (2.2) the name 'Obama' seems tied to a specific person, and so, in our relational Kripke structure for this sentence, the same person would be tall in every world that Alice considers possible. For Alice, 'Obama' is a *rigid designator* (to use the philosophical jargon), while 'the

U.S. president' is a *non-rigid designator*. (Cf. [9] for a good overview of the concept of "rigid designator.")

It is easy to model cases like this one in which it is clear whether a name is rigid or non-rigid; we just include the denotations of the rigid names (in this example, 'Obama') in the domain, and we add a predicate for each non-rigid designator. Let $M = (\mathcal{W}, B_a, \pi)$ be a Kripke structure where B_a is Alice's belief-relation. For each world w in $\mathcal{W}$, the domain $w.Dom$ contains people, and there are a set $w.T$ of all tall people and a set $w.P$ containing just the person who is the president. Let w_1 be a world in which (2.2) is true and w_2 be one in which (2.3) is true. Then Obama is in both $w.Dom$ and $w.T$ for any w such that $(w_1, w) \in B_a$, and $w.P \subseteq w.T$ for any w such that $(w_2, w) \in B_a$. This model can be concisely described with a formal language. For all w in $\mathcal{W}$, let $\pi(w)(\text{`Tall'}) = w.T$, $\pi(w)(\text{`President'}) = w.P$, and $\pi(w)(\text{`}o\text{'}) =$ Obama. We now have the following (V_0 is an empty valuation):

$$(M, w_1, V_0) \vDash \text{`}\mathbb{B}_a \operatorname{Tall}(o)\text{'}$$
$$(M, w_2, V_0) \vDash \text{`}\mathbb{B}_a \forall x(\operatorname{President}(x) \rightarrow \operatorname{Tall}(x))\text{'}$$

There are still cases involving non-rigid names that are difficult to describe with this logic. (In [10], Grove makes the following points, and the following example is derived from there.) Suppose we have a system of autonomous, mobile agents, and an agent a breaks down and broadcasts a "help" message that it hopes will be received by the helper agent h. These agents do not have a standard naming system, and so the "help" message contains directions to a's location. However, the transmission medium is quite unreliable and parts of the message may have been lost or corrupted; a will keep broadcasting the "help" message until it receives an acknowledgement from h. How should we formally express the condition under which a should stop broadcasting the "help" message? Consider:

$$\mathbb{B}_h \operatorname{NeedsHelp}(a) \tag{2.4}$$

Informally, 'h believes that a needs help'. Is this good enough? Or is it possible that (2.4) is true but a should continue calling for help? Suppose the message originally said 'The agent in room 22 needs help' but was corrupted on its way to h, so that h ended up seeing 'The agent in room 2 needs help'. Unaware that the message was corrupted, h sends the acknowledgement and proceeds toward room 2. Clearly a should continue sending the message, and yet (2.4) is

in some sense true: h believes 'The agent that just sent that message needs help' and the name 'the agent that just sent that message' denotes a.

The problem with (2.4) is that it does not say *how* h refers to a; it just says that h refers to a *somehow*. What we need is a formula that says something like 'h can find a by following some list of directions, and h believes that the agent that it can find by following those directions needs help'. Let the predicate '$\mathrm{GoToRoom22}$' denote the property whereby an agent can be found by h by going to room 22: then we can replace (2.4) with this:

$$\mathbb{B}_h \, \forall x \big(\mathrm{GoToRoom22}(x) \rightarrow \mathrm{NeedsHelp}(x) \big) \tag{2.5}$$

The English equivalent to (2.5) is 'h believes that the agent in room 22 needs help', where 'the agent in room 22 is a non-rigid designator'. If a were to believe (2.5), it would stop broadcasting the "help" message, because (2.5) says that h knows all it needs to in order for it to find and help a. However, although (2.5) is fine for our current example, we would need a different predicate instead of '$\mathrm{GoToRoom22}$' whenever a is in a different location. Moreover, we cannot generalize (2.5) without a higher-order logic that can quantify over predicates; for example, how would we formalize 'h believes that, for any room r, the agent in r needs help'? So we see that there are situations involving non-rigid names that conventional quantified modal logic cannot handle.

2.3.2 Dealing with Quantification

In the above examples, I have been assuming that different worlds can have different domains. In fact, this is not an obvious assumption; many quantified modal logics make the *common-domain assumption*, which says that all the worlds have the same domain. It turns out that, with the common-domain assumption, it is easier to adapt standard inference rules for quantifiers to quantified modal logic than it is if we allow worlds to have different domains. Logics with the common-domain assumption are characterized by the validity of what is called 'the Barcan Formula' (cf. [11]), which is really a class of formulas:

$$\ulcorner \forall \alpha \, \Xi \, \phi \rightarrow \Xi \, \forall \alpha \, \phi \urcorner$$

('α' stands for a variable, 'Ξ' a modal operator, and 'ϕ' a formula in which α may be free.) For a quantified modal logic of belief, the common-domain assumption implies that in every world everybody knows who or what exists.

The main purpose for allowing different domains is to represent cases in which the believer is unsure (or incorrect) about who or what actually exists. One difficulty with this approach is dealing with formulas like the following:

$$\forall x \, \mathbb{B}_a \, \mathrm{Tall}(x)$$

Suppose this formula is true at world w. This seems to say that Alice thinks that everyone is tall. In fact, it effectively says more than this, because it forces the domains of all the worlds that Alice thinks possible to be supersets of w's domain. For, according to the definition of '$\models$' given earlier, the formula is true at w if and only if, for any member i of $w.Dom$, '$\mathbb{B}_a \, \mathrm{Tall}(x)$' is true at w when 'x' is taken to denote i; but this means that, for any world w' that Alice thinks possible at w, '$\mathrm{Tall}(x)$' is true when 'x' is taken to denote i, and this in turn means that i is a member of $w'.T$ (the set of all tall people at w') and thus a member of $w'.Dom$. It is difficult to design quantified modal logics with varying domains, even though having varying domains makes sense for many applications.

Quantified modal logic presents a minefield of issues, including philosophical ones, but it also provides opportunities for formally dealing with naming. In Chapter 3 we will look at Grove's attempt ([10]) to take advantage of them.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Literature Review

This thesis builds on works from three areas of inquiry: computer security, formal logic, and philosophy of language and logic. Most relevant are works about belief in security protocols, such as Rangan's [12]; BAN Logic and related logics ([13], [14], [15], [16], [17], [18], and [19]); and Yahalom *et al.*'s [20]. [12] and [20] are particularly relevant in that they are motivated by the idea that it is important to understand a protocol's trust-assumptions. BAN Logic, [12], and [20] are discussed in more detail below.

Another related area is "trust management": security systems that explicitly represent and reason about trust in order to enforce security policies. The term 'trust management' seems to have been introduced by Blaze *et al.* in [21]. Recent works in this area include Li *et al.*'s [22] and Guttman *et al.*'s [23]. Although they do not use the term 'trust management', Appel *et al.* present a similar system in [24], which is discussed below.

The mathematical model that we begin later in this thesis (Chapter 5) uses two important modeling methods: modal logic with possible-worlds semantics, and multi-agent systems. The best reference for combining these two methods is Fagin *et al.*'s [6]. This thesis's pursuit of a mathematical model was inspired by the potential usefulness of a quantified modal logic of naming such as that presented in Grove's [10], which is presented below.

Perhaps surprisingly, quantified modal logic is closely related to philosophical questions about naming and modes of truth (e.g., belief, necessity) that go back to Frege's 1892 paper [25], which is perhaps the first place where the strange logical properties of modalities and names are discussed. Quine first presented his approach to these questions in [26]. Barcan's [11], in which she presents a quantified modal logic, and Quine's response to Barcan in [27] clearly show how these questions are relevant to quantified modal logic. (We already saw some of these issues in Section 2.3.) Quine expands his approach in [28]. In [29], Kaplan compares the approaches of Frege and Quine and then attempts to answer these questions in terms of different kinds of names. Later (Chapter 4), we will find Kaplan's ideas about kinds of names quite useful. As a way of also introducing Frege's and Quine's approaches, Kaplan's comparison of them is presented very briefly below.

It is of course not possible to review in detail all previous works related to the problem and

approach presented in this thesis. Instead, I have chosen just a few works that, I hope, will put the present work in context.

3.1 Rangan's "Axiomatic Basis of Trust"

The goals and method of this thesis are very similar to those of the approach of P. Venkat Rangan presented in the 1988 paper "An Axiomatic Basis of Trust in Distributed Systems" ([12]). Rangan believed that security protocols for the then-emerging distributed systems of computers would have to deal with the absence of a single universally trusted source of information, in order for the independent organizations and people who own those computers to get any benefit from interacting with each other. For any security protocol, it is necessary to understand the particular kinds of trust it uses (or assumes), in order for us to decide whether that protocol should be used for a given situation.

As we do in Chapter 5, Rangan models trust using a modal logic of belief and the multi-agent system framework. He applies possible-worlds semantics to the modal logic and interprets possible global states of a system as possible worlds. For each participant, belief is represented with a transitive, Euclidean, and serial binary relation over the possible global states. (A binary relation R over a set S is "transitive" if and only if, for any elements a, b, and c of S, if $(a, b) \in R$ and $(b, c) \in R$ then $(a, c) \in R$. R is "Euclidean" if and only if, for any elements a, b, and c of S, if $(a, b) \in R$ and $(a, c) \in R$ then $(b, c) \in R$. R is "serial" if and only if, for any element a of S, there exists an element b of S such that $(a, b) \in R$.) A *trust* is a belief held by a participant before and throughout a run of a protocol — in other words, an assumption. A participant's trusts are represented as constraints on the belief-relation, but the rest of the beliefs (that is, beliefs acquired as a result of a run of the protocol) are determined solely by the participant's local state.

Another similarity between this thesis and Rangan's work is in his approach to discovering the trusts of a protocol. The security goals of a given system are encoded as formulas in the modal logic; we then must look for other formulas (asserting some beliefs on the part of some participants) such that, when they are added as additional axioms to the logic, the goal formulas can be derived.

In our attempt to make a mathematical model of security protocols, we do not use Rangan's approach. His approach does not account for the use of names in participants' beliefs, and thus it is not possible for it to deal with cases of relative or ambiguous names. Also, in his

multi-agent system model, the meanings of messages that participants send to each other are explicitly represented by formulas contained in the message, which prevents participants from interpreting messages differently. Consequently, we must decide on the "correct" interpretation of a protocol's messages before we can model it. In contrast, in the approach presented in Chapter 5 we represent participants' interpretations of messages as assumptions made by these participants.

3.2 BAN Logic

In the late 1980s and early 1990s, Michael Burrows, Martín Abadi, and Roger Needham developed a formal logic for analyzing authentication protocols that has been very influential and has come to be called 'BAN Logic'. There is no definitive version of BAN Logic; it was presented with variations in multiple papers in 1989 and 1990 ([13], [30], and [16]). It quickly received much attention (e.g., [17], [31], [32], and [18]). In response to some criticism (Syverson's [32] provides a good analysis of the issues), Abadi and Mark R. Tuttle presented ([19]) a revised version with a new semantics. I present the revised version here.

The purpose of the logic is to enable analysis of authentication protocols in terms of the beliefs of the participants. The goal of a particular protocol would be defined as the holding of particular beliefs by the participants, and the analysis would consist of showing the effects of the protocol's messages on the participants' initial and subsequent beliefs. Therefore, BAN Logic's language focuses on belief, message-passing, and the construction of messages with cryptographic operations. The semantics is based on a model of a distributed system as a multi-agent system in which the set of agents contains all the principals along with a special agent P_e representing the environment. Each principal's state contains a record of the actions that the principal has performed (a "local history") and the set of keys that the principal holds (a "key set"). The environment's state includes a record of *all* the actions performed (a "global history"), a set of keys, and, for each principal P_i, a buffer containing all the messages sent to P_i but not yet received by P_i. A principal changes its state by executing actions, but only its own state and the environment's state can change. The possible actions include sending a message, receiving a message, and receiving a previously unknown key. Runs are required to satisfy certain reasonable constraints, such as the constraint that a message must be sent before it is received.

An important feature of the model is the division of a run into two "epochs," the past and the present. The states in a run are assigned consecutive integers representing times. The negative

integers represent times in the past, while nonnegative integers represent times in the present. In this way, BAN Logic can model recency of message-transmissions, which is important when modeling nonces.

A protocol is modeled as a set of functions $\{A_1, \ldots, A_n, A_e\}$, one for each agent (the principals and the environment), that map local states to actions. If agent i is in local state s, then i must perform $A_i(s)$.

Belief is modeled with the possible-worlds approach, using points as possible worlds.

Based on this model, BAN Logic defines formulas such as the following ('ϕ' is a metavariable for formulas):

- 'A **says** X' ('A sends the message X (in the present)')
- 'A **said** X' ('A sent the message X')
- '$\mathbf{fresh}(X)$' ('The message X was sent for the first time in the present')
- 'A **has** K' ('A has key K')
- '$A \overset{K}{\leftrightarrow} B$' ('$A$ and B share the key K')
- '$A \overset{X}{\rightleftharpoons} B$' ('$A$ and B share the secret message X')
- '$\ulcorner A$ **believes** $\phi \urcorner$' ('$\ulcorner A$ believes that $\phi \urcorner$')
- '$\ulcorner A$ **controls** $\phi \urcorner$' ('$\ulcorner$Whenever A says that ϕ, then $\phi \urcorner$')

The definitions of '**says**' and '**fresh**' distinguish between the two epochs. '**has**' is defined as a condition on a principal's set of known keys. '$\leftrightarrow$' and '$\rightleftharpoons$' are both defined as conditions on the messages sent in a run (in both epochs). If K is a shared key ('$\overset{K}{\leftrightarrow}$') between two principals, then the only way another principal could send a message encrypted with K is by first receiving that encrypted message from someone else. Similarly, if X is a shared secret ('$\overset{X}{\rightleftharpoons}$') between two principals, then the only way another principal could send a message containing X is by first receiving that message from someone else.

The papers in which BAN Logic was presented have been very influential. These papers start with a strong and important claim: that the goal of authentication is to cause certain beliefs in

the participants. They then present an intuitive formalism for reasoning about how the messages exchanged in authentication protocols affect participants' beliefs. The language is very concise, the rules are intuitive, and the focus of the analysis — belief — is explicitly represented. It has some problems, some of which I discuss below, but in many ways it is an example of what a formal method should be.

3.2.1 Messages and Names

We do not use BAN Logic in this thesis mostly because it assumes that all the principals have unique and commonly known names. If in some protocol (e.g., AdultVerify, presented in Section 4.4) a principal holds beliefs about another principal without knowing the latter's proper name, then this protocol cannot be analyzed with BAN Logic.

There are other problems with BAN Logic that should be briefly discussed. These problems are in its treatment of messages. In the multi-agent system used as BAN Logic's model, principals can send keys, principals' names, "primitive propositions" ([19, p. 207]) (whatever those are), and "things like nonces" (ibid.) to each other as messages, as well as the results of applying certain operations to messages: concatenation, encryption, and inclusion of a shared secret. The problem is that they can also send BAN Logic formulas; BAN Logic's language is thus a part of its model! As we saw, this also occurs in Rangan's model, and it is a problem for the same reason: a principal's interpretation of concrete messages (that is, bit-strings) should be our (the analysts') object of study; we should not interpret the concrete messages ourselves.

3.3 Yahalom, Klein, and Beth's "Trust Relationships"

The basis of the 1993 paper "Trust Relationships in Secure Systems - A Distributed Authentication Perspective" ([20]), by Raphael Yahalom, Birgit Klein, and Thomas Beth, is the authors' observation that communication protocols rely on the participants' trust-relationships just as much as they rely on cryptographic mechanisms and shared secrets, and that there are a variety of trust-relationships that protocols can take advantage of. The authors examined protocols that are used to establish secure communication over the Internet (e.g., Kerberos ([33])) and found seven classes of trust-relationships that a user might need to hold with regard to a certification authority (CA) in order to use such a protocol. Each class is related to a specific kind of service that CAs typically provide, such as correctly asserting that a name belongs to a principal, providing high-quality cryptographic keys, and maintaining a clock that can be used to ensure freshness of messages.

The significance of "Trust Relationships" for this thesis is the fact that Yahalom *et al.* were motivated by the following observations: (1) that trust has an important role in communication protocols, (2) that there are multiple kinds of trust, and (3) that different protocols require different kinds of trust. In the authors' words:

> Explicit specification of such relationships [i.e., trust] may help to determine whether a certain protocol or certain initial assumptions are realistic in a given environment. [...] Explicit analysis of required trust relationships can be used as a criterion for comparing different protocols (e.g. which ones have weaker trust requirements) as well as to extend protocols in a way which provides more flexibility — adapting them to varying environments with different trust relationships. [p. 152]

Yahalom *et al.* thus made some of the same observations about assumptions in security protocols that we harp on in this thesis (Chapter 4 and Chapter 6). Unfortunately, they did not consider trust classes involving names — that is, trust classes in which a principal is trusted to use or generate names in some particular way. They also did not consider how to analyze trust when there is no universal naming system.

3.4 Appel and Felten's "Proof-carrying Authentication"

In 1999, Andrew W. Appel and Edward W. Felton presented ([24]) a formalism that can express many of the things that BAN Logic can express. (I will refer to this formalism as 'Appel-Felton Logic'.) Appel-Felton Logic is a non-modal, higher-order logic. The key feature of this logic is that it models each principal as a predicate that identifies the formulas that that principal believes (or, in their words, "admits as true"). Such a predicate is called 'a worldview'.

Identifiers in Appel-Felton Logic can refer to strings, formulas, functions, and predicates, including worldviews; all of these sorts can be quantified over. Atomic formulas have the form $\ulcorner\omega(\phi)\urcorner$, where ω is a variable denoting a worldview and ϕ is a formula or a variable denoting a formula; $\ulcorner\omega(\phi)\urcorner$ means that the formula (denoted by) ϕ is true in the worldview denoted by ω. (This logic confusing using and mentioning because it is higher-order.) "Principals" are a special type of worldview: worldview P is a principal if and only if the following are true

(p. 54):

$$\forall F \,.\, F \longrightarrow P(F)$$
$$\forall F \forall G \,.\, \big(P(F) \wedge P(F \rightarrow G)\big) \longrightarrow P(G)$$

('F' and 'G' range over formulas.) The logic includes a primitive function called '$\mathcal{N}$' that constructs principals from strings. In this way, Appel-Felton Logic distinguishes between principals and their names. The logic does not have axioms, only inference rules. (Interestingly, one of the inference-rules (n_is_prin, p. 55) implies that $\mathcal{N}$ is a total function.)

Unlike BAN Logic, Appel-Felton Logic was not designed as a protocol-analysis tool. Instead, the intended use of this logic is as an implementation-language for various methods of distributed authentication — for example, the authors mention SPKI ([34], [35]) and X.509 ([36]). The fact that this logic is higher-order enables one to define specific predicates for a particular authentication method — in other words, it can be used to implement a domain-specific language for a particular type of authentication.

For example (pp. 56–57), we could define some predicates for authentication methods that use public-key cryptography:

$$\ulcorner \omega \text{ says } \phi \urcorner \stackrel{\text{def}}{:=} \ulcorner \omega(\phi) \urcorner$$
$$\ulcorner \omega_1 \text{ speaksfor } \omega_2 \urcorner \stackrel{\text{def}}{:=} \ulcorner \forall x \,.\, \omega_1 \text{ says } x \longrightarrow \omega_2 \text{ says } x \urcorner$$
$$\ulcorner (\text{cert } \omega_1 \, \sigma \, \omega_2) \urcorner \stackrel{\text{def}}{:=} \ulcorner \omega_1 \text{ says } (\mathcal{N}(\sigma) \text{ speaksfor } \omega_2) \urcorner \quad (\sigma \text{ is an identifier of sort string})$$

Thus, '(cert C K_a A)' can represent the fact that the certification authority C has bound the public key K_a to A. Notice how K_a is used (appropriately) as a string. The assumption is that, if the server receives a message containing the formula F signed with A's private key, it will infer '$\mathcal{N}(K_a)$ says F'. The function $\mathcal{N}$ reflects the assumption that only one person has the corresponding private key and can thus cause $\mathcal{N}(K_a)$ to "speak."

Appel-Felton Logic shows how a higher-order language can be used to create domain-specific languages which, in turn, can be used to enforce security policies. A key idea in their paper is that requests should include proofs of the conditions which must hold (according to some security policy) in order for the requests to be satisfied. This enables security policies to be specified declaratively (in the domain-specific language) while still not requiring the server to

generate the proofs itself but merely to verify them.

Appel-Felton Logic is powerful, but it cannot deal with some of the issues of naming that are of concern in this thesis — for example, the possibility that a name does not actually refer to anything. Also, it does not have a model with which we can justify the inference rules.

3.5 Kaplan's "Quantifying In"

David Kaplan takes up the issues raised by Quine ([28]) about the logical properties of what Quine calls "referentially opaque" contexts. He contrasts Quine's analysis with what he presents as Frege's ([25]), and argues that Frege's provides a better explanation. Briefly, Kaplan describes Quine's analysis as treating names inside opaque contexts as meaningless, since they do not have the same logical properties as they do outside such contexts. By contrast, Kaplan presents Frege's analysis as treating names inside opaque contexts as just as meaningful as in normal contexts, but having a *different* meaning from the normal one. For example, consider the following sentence:

Ortcutt is a spy, and Ralph believes that Ortcutt is a spy. (3.1)

This sentence's opaque context is everything to the right of 'that'. For Quine (according to Kaplan), the second occurrence of 'Ortcutt' does not mean anything, at least as far as logic is concerned. But for Frege, the second occurrence of 'Ortcutt' is just as referential as the first, but the two occurrences do not have the same meaning. The second one refers to something other than Ortcutt, something more *mental*. Kaplan shows that with Frege's analysis we can explain why, for example, substitution fails when it crosses the border of opaque contexts: we are substituting an expression that means something different than the one being replaced.

Kaplan hypothesizes that these strange mental entities that are referred to in opaque contexts are linguistic — specifically, expressions. So, when we quantify into opaque contexts, we must be sure to only quantify over expressions, as in the following:

$$\exists\alpha \text{ . Ralph believes } \ulcorner \alpha \text{ is a spy} \urcorner$$

('α' ranges over expressions.)

This still leaves us unsure about what if any relation between Ralph and Ortcutt is implied

by (3.1), since it now seems that belief is just a relation between the believer and a sentence. Kaplan's main contribution to Frege's approach in this paper is to shed some light on the very tricky relations between believer, belief, and the things the belief is about, and to show that there are different kinds of names that have different logical properties in opaque contexts. Consider the difference between the following:

Ralph believes that Ortcutt is a spy.

Ralph believes that the shortest spy is a spy.

Assuming that 'Ortcutt' is the name of someone Ralph knows and that there is a shortest spy, both of these sentences have names in opaque contexts that denote people, but, as Quine would say, the FBI would only be interested in the first one. This is just a simple example of Kaplan's distinctions between kinds of names based on their causal relations with their subjects and objects and their descriptive contents; we will find this distinction useful later (Subsection 4.3.2). At the end, he picks a kind of name ("vivid names") that he claims guarantees a connection between the subject and object strong enough to allow quantification and substitution into opaque contexts.

3.6 Grove's Logic

In 1995, Adam Grove presented ([10]) a quantified modal logic of knowledge for describing systems of multiple interacting agents. Grove's principal goal for the logic was to account for the names that agents use to refer to each other, but he also includes features to represent a kind of *subjective* knowledge. This logic is an extension of a propositional logic with similar goals that Grove and Joseph Halpern presented ([37]) in 1993. Because of the quantified logic's explicit treatment of names, it is the inspiration for the mathematical model begun later (Chapter 5) in this thesis.

3.6.1 The Issues

As Grove sees it, a formal model of knowledge that accounts for naming should deal with two issues. The first issue is about the scope of non-rigid names used in descriptions of agents' knowledge. To explain, Grove gives an example:

> ... consider an unreliable network in which some processes may have failed. Let the formula ϕ stand for "p_1 knows that all correct processes received p_1's last message".

> Suppose that, in fact, the correct processes are p_1, p_2, and p_3, and p_1 knows that all three of these processes actually did receive the message. [10, p. 313]

Whether ϕ is true depends on the meaning of the name 'all correct processes'. We know that p_1, p_2, and p_3 are all the correct processes, and so if we interpret this name from our (i.e., an objective) perspective, then ϕ is true. But suppose that p_1 thinks that there is some other process, p_4, that is also correct: then from p_1's perspective, ϕ is false "(and, perhaps, may spend more time — fruitlessly — waiting for additional acknowledgments)" (ibid.). Each perspective introduces a *scope* in which names like 'all correct processes' are interpreted. p_1's perspective is the *innermost* scope for this name, whereas the *outermost* scope is the real world — that is, the world in which the sentence is said to hold. The issue is that different scopes can be useful for different names and sentences, and so it is desirable that a formal model of knowledge enables names to be interpreted at arbitrary scopes. In most quantified modal logics with non-rigid constants, 'all correct processes' would be re-evaluated in each of the worlds that p_1 considers possible, thus giving it an innermost-scope interpretation.

Suppose that we interpret 'all correct processes' using the outermost scope, which would make ϕ true. We thus know that the knowledge that ϕ attributes to p_1 is about p_1, p_2, and p_3, but we do not know *with what name or names* p_1 refers to p_1, p_2, and p_3. There are many possibilities: for example, p_1 may use the name 'all correct processes' to refer to them collectively, or it may use a different name for each of them. This is the issue that Grove calls 'multiple ways of referring'.

3.6.2 The Logic

To build a logic that addressed these issues, Grove apparently found it necessary to have it deal with subjective knowledge as well, as in 'the agent who just sent *me* a message' and '*I* know that Bob needs help'. To do this, he used a variant of possible-worlds semantics based on ideas from the work of David Lewis ([38]) and Yves Lesperance ([39]). In his model, a formula is interpreted not in terms of a world but in terms of a world-agent pair, and instead of having multiple knowledge-relations K_i over worlds, it has just one knowledge-relation K over world-agent pairs. Briefly, a world-agent pair (w, a) represents the world w from the point of view of agent a; if $((w, a), (w', a')) \in K$, then a in w thinks that w' could be the real world and that he could be a' in w'. It is difficult to get an intuition for this, and I will not do it much justice here — for details, cf. Grove's paper as well as Lesperance's. The effect on the logic's language is that there is a term '$\mathbf{me}$' such that, if $(w, a) \vDash \phi$, then '$\mathbf{me}$' denotes a wherever it occurs in ϕ

outside the scope of a modal operator. For example, $(w, p_1) \vDash$ '$\mathrm{NeedHelp}(\mathbf{me})$' if and only if p_1 needs help in world w.

To deal with names, Grove made his logic *multi-sorted*, with two sorts: agents and names. This means that all identifiers (i.e., all constants and variables) in the language fall into exactly one of these sorts. All terms except '$\mathbf{me}$' consist of a single letter; lowercase terms are of sort agent, and capital terms are of sort name. '$\mathbf{me}$' is also of sort agent. Correspondingly, in the semantics each world has two domains: a domain of agents and a domain of names. Worlds can contain different agents, but every world must have the same name-domain. Every world also has a special ternary predicate, denoted by 'In' in the language, that represents the relationship between a name, the agent who uses the name, and the agent denoted by the name.

A key principle of Grove's logic is that, when one agent reasons about another, it does so using some name. This principle is reflected in the logic's syntax in the form of a restriction that bans formulas such as '$\exists x\, \mathbb{K}_{\mathbf{me}}\, P(x)$' in which a quantifier over an agent variable binds into the scope of a modal operator. As Grove says, "Such formulas assert knowledge about someone (some x) but they do not say anything about *how* this agent is actually being referred to (by $\mathbf{me}$, or whoever is doing the reasoning)" ([10, p. 329]).

Grove gives (p. 326) the following formula to show off his logic:

$$\exists x\Big(\mathrm{Talking}(\mathbf{me}, x) \wedge \exists X\big(\mathrm{Location}(X) \wedge \mathsf{In}(\mathbf{me}, x, X) \\ \wedge \mathbb{K}_{\mathbf{me}}(\forall y : \mathsf{In}(\mathbf{me}, y, X) \rightarrow \mathrm{Tall}(y))\big)\Big)$$

He interprets it as follows:

> I ($\mathbf{me}$) am talking to some agent (x), x's position relative to me is captured by some "location-type" name X (for instance, X might be "in front of"), and I know that the agent in that location is tall. [pp. 326–327]

To see how the logic confronts the issues mentioned above, consider again the example of processes in an unreliable network. Suppose that they refer to each other using numbers, thus:

#1 knows that #5 knows that #1 needs help.

The logic should enable us to express all the different meanings of this sentence corresponding to different scopes for the names. The innermost-scope interpretation is expressed thus:

$$\forall x\Big(\mathsf{In}(\mathbf{me}, x, \#1) \rightarrow \mathbb{K}_x \forall y\big(\mathsf{In}(\mathbf{me}, y, \#5) \rightarrow \mathbb{K}_y \forall z(\mathsf{In}(\mathbf{me}, z, \#1) \rightarrow \mathrm{NeedsHelp}(z))\big)\Big) \quad (3.2)$$

Each occurrence of '$\mathbf{me}$' potentially denotes a different agent; the denotation is determined by the scope, and scope-boundaries are marked by occurrences of modal operators. For example, the last '$\mathbf{me}$' is in the scope of '$\mathbb{K}_y$' and thus has the same denotation as 'y'; the second '$\mathbf{me}$' has the same denotation as 'x'. The first '$\mathbf{me}$' denotes the agent in whatever world-agent pair this formula is interpreted for (which we have not specified). Notice that the second instance of '#1' does not necessarily refer to the same agent as the first instance does.

There are multiple possible outer-scope interpretations. We will formalize one in which both instances of '#1' refer to the same agent:

$$\forall x\Big(\mathsf{In}(\mathbf{me}, x, \#1) \rightarrow \mathbb{K}_x \forall y\big(\mathsf{In}(\mathbf{me}, y, \#5) \rightarrow \exists X\big(\mathsf{In}(y, \mathbf{me}, X) \wedge \mathbb{K}_y \forall z(\mathsf{In}(\mathbf{me}, z, X) \rightarrow \mathrm{NeedsHelp}(z))\big)\big)\Big) \quad (3.3)$$

Because this outer-scope interpretation does not say how "y" (that is, the agent called '#5') refers to x — just that it *does* refer to x — this formula must hypothesize some name ('$\exists X$') in order to describe y's knowledge.

(3.3) also shows how this logic resolves the issue of multiple ways of referring: after '$\exists X$', the formula could have included material that says more about X — for example, '$\mathrm{IpAddress}(X)$'.

3.6.3 Critique

I will mention one issue that should be resolved before employing the logic: the role of constants other than '$\mathbf{me}$' — namely, 'a', 'b', 'A', 'B', etc. In the semantics of Grove's logic as well as traditional quantified modal logic, constants' denotations can vary from world to world — effectively, they always have innermost scope. Since Grove's logic provides a more rigorous way of expressing innermost-scope interpretations, what purpose do constants serve in the language? Interestingly, in most of Grove's examples of formulas in his language the only agent constant is '$\mathbf{me}$'. More importantly, the denotations of name constants also vary by world, and this seems to undermine the usefulness of formulas like (3.2), because otherwise the rigidity of

the name constants (e.g., '#1') would enable agents to reason about each other's knowledge.

As mentioned above, Grove's logic was very inspirational for the mathematical model begun in Chapter 5. In fact, our goal in that chapter will be to create a model that could form a semantics of a logic like Grove's, adapted to belief. The most compelling aspect of Grove's logic is that it explicitly deals with names as well as agents, and this means that it can be used to describe different *kinds* of names (such as the kinds mentioned by Kaplan). The ability to model subjective knowledge has some interesting potential as well: for example, if agent c receives some random message M, what better way is there to represent c's name for the sender than with something like 'the agent who just sent *me* M'?

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Survey of Beliefs in Security Protocols

In this chapter, we will consider five example protocols. For each example, we will choose one of the principals and then consider the following questions:

- What belief is the principal supposed to have at the end of a run of the protocol?
- What assumptions is the principal supposed to make as a participant in the protocol?

Before we consider the examples, we must have a general idea of the relationships between a protocol's goals and the principals' beliefs.

4.1 General Model of Security Protocols' Goals

A security protocol is designed to enforce at least some parts of a security policy, which usually includes goals that fit into the venerable "CIA Triad" — Confidentiality, Integrity, and Availability. The parts of the security policy that a protocol is supposed to enforce constitute the protocol's *security goals*. We consider a security protocol to be *broken* when we discover that there is a (perhaps hypothetical) case in which the protocol is used correctly but does not achieve its security goals.

Security goals can often be stated in terms of events or behaviors rather than beliefs or knowledge (e.g., confidentiality of some message M can be defined as the inability of an unauthorized principal to send M). For example, the strand space method ([40]) analyzes protocols purely in terms of possible sequences of events. To the extent that a protocol's security goals do not involve belief, whether that protocol is broken is independent of what any principal believes.

However, security protocols clearly do have goals that involve belief. For example, as Paul Syverson ([32]) observed, a secure key is not much good to me if I do not believe that it is secure. In general, for each role of a security protocol, there is a set of *goal beliefs* such that whoever is playing that role is supposed to end up with those beliefs at the end of a run of the protocol. In other words, one of the goals of a security protocol is that, at the end of a run, each principal has acquired its goal beliefs. Security goals and goal beliefs are distinct but related: a

role's goal belief expresses the security goals that are important to whoever is playing that role. If those security goals have not been achieved, then the goal belief will be false.

During a run, principals are not expected to acquire their goal beliefs out of thin air; rather, they should acquire them as a result of their participation in the run and of beliefs that they held before the run began. For example, a principal — Bob — with knowledge of asymmetric cryptography and the belief that K is a public key can conclude, after observing the message $M' = \{\!|M|\!\}_{K^{-1}}$, that M' was made by someone who possessed K^{-1}. If Bob also believes that Alice is the only principal who possesses K^{-1}, then he can also conclude that Alice created M'. If Bob also has certain beliefs about the intended interpretation of M' in the context of the protocol, then he can conclude that Alice wants to communicate some idea or proposition that is encoded in M. In general, security-protocol designers assume that a participant starts out with a particular set of beliefs, and these constitute his role's *required assumptions*; designers construct the protocol so that the participant will observe a sequence of events that causes him to infer his role's goal beliefs.

In this rest of this chapter, we will investigate the required assumptions of some example protocols.

4.2 Message Authentication with Digital Signatures

Suppose Alice works at BigBank, and that BigBank signs all important internal email messages with private key K_{BB}^{-1} and gives its employees its public key K_{BB} so that they can authenticate the messages. Whatever the particular protocol used for signing email messages, it has the form shown in Figure 4.1.

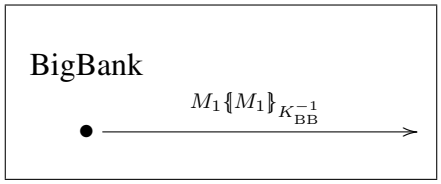

Figure 4.1: A Simple Email Signature Protocol for BigBank. M_1 is an unsigned email message. M_1 also occurs outside the signature so that the recipient can verify that he or she used the correct public key to check the signature.

Suppose the run shown in Figure 4.1 occurs — that is, Alice receives $M_1\{\!|M_1|\!\}_{K_{\mathrm{BB}}^{-1}}$. Because she has K_{BB}, she knows that the term following the first instance of M_1 is the result of encrypting M_1 with K_{BB}^{-1}. For this run of the protocol, its goal is to convince Alice of the following:

Goal Belief 4.2.1 (For Alice)**.** $M_1\{\!|M_1|\!\}_{K_{\mathrm{BB}}^{-1}}$ *was sent by BigBank.*

Alice must start with some assumptions in order to infer this belief. There are many possible assumptions that would work, but it is reasonable to assume that Alice's role in the protocol is intended to be filled by someone making assumptions like the following:

Assumption Set 4.2.1 (For Alice)**.**

1. *For any message* M *and public key* K*, if I receive* $M\{\!|M|\!\}_{K^{-1}}$ *then someone with* K^{-1} *sent it.*

2. *Only BigBank has* K_{BB}^{-1}*.*

There are some important differences between these two assumptions. The first assumption is about the protocol's cryptography, and if it is ever false we would consider the protocol to be broken. But we would not necessarily consider it to be broken if the second assumption is false.

Now suppose that, instead of an employee of BigBank, Alice is a customer. K_{BB} is now a public key that BigBank publishes in a certificate and that it uses for communication with customers (e.g., on a website). In general, along with a public key, a certificate contains identifying information such as a name, an address, and an email address; we will call such a collection of information 'an identifier'. For any identifier N and public keys K_1 and K_2, we will use '$\mathrm{sigcert}(N, K_1, K_2)$' to denote the certificate containing N and K_1 and signed with K_2^{-1}. Let N_{BB} be the identifier on BigBank's certificate. Suppose that there is a certification authority Auth whose public key is K_{Auth} and who has signed BigBank's certificate to make $\mathrm{sigcert}(N_{\mathrm{BB}}, K_{\mathrm{BB}}, K_{\mathrm{Auth}})$. Finally, suppose that Alice has K_{Auth}. The protocol (whatever the details) still has a simple form, shown in Figure 4.2.

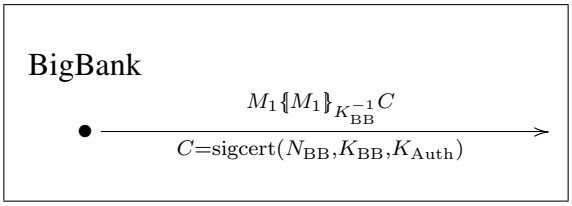

Figure 4.2: A Simple Email Signature Protocol for BigBank, with Certificate Exchange. M_1 is an unsigned email message.

Suppose such a run occurs. This run has a similar goal belief to the first one, but for this kind of protocol (specifically, message-signature protocols relying on a public-key infrastructure (PKI)), Alice is supposed to make assumptions like the following:

Assumption Set 4.2.2 (For Alice)**.**

1. *For any message M and public key K, if I receive $M\{\!|M|\!\}_{K^{-1}}$ then someone with K^{-1} sent it.*
2. *For any identifier N and public key K, if I receive $\mathrm{sigcert}(N, K, K_{\mathrm{Auth}})$ then Auth believes that only the principal identified by N has K^{-1}.*
3. *For any identifier N and public key K, if Auth believes that only the principal identified by N has K^{-1}, then only the principal identified by N has K^{-1}.*
4. *The principal identified by N_{BB} is BigBank.*

(Note that receiving $M\{\!|M|\!\}_{K^{-1}}C$ counts as receiving $M\{\!|M|\!\}_{K^{-1}}$.) The first assumption is about the protocol's cryptography. The second assumption says that $\mathrm{sigcert}(N, K, K_{\mathrm{Auth}})$ is Auth's way of saying that it has a certain belief, and that Auth is telling the truth when it says this. In other words, the second assumption says that Auth is *honest*. The third assumption says that Auth's belief is true — in other words, that Auth is *competent*. We could say that these two assumptions constitute Alice's *trust* in Auth. The fourth assumption is about the meaning of the name N_{BB}. The goal is achieved when Alice applies these assumptions to infer that Auth believes that only the principal identified by N_{BB} has K_{BB}^{-1}, hence that only the principal identified by N_{BB} has K_{BB}^{-1}, hence that only BigBank has K_{BB}^{-1}, and finally that BigBank sent $M_1\{\!|M_1|\!\}_{K_{\mathrm{BB}}^{-1}}$.

In this example, we have identified two categories of assumptions — assumptions of honesty and assumptions of competence — that constitute *trust* in another principal. In general, many security protocols (and all protocols that use "trusted third parties" like Auth) will have such required assumptions, which the principals must accept in order to achieve their goals. We will look at the required assumptions of three more protocols, but first we should consider what these assumptions tell us about a protocol.

It may be that the cryptography used in this protocol is perfect; nevertheless, if the last three assumptions are false, Alice could still end up in an insecure position. For example, the protocol

can guarantee that only someone with K_{BB}^{-1} sent $M\{\!|M|\!\}_{K_{\mathrm{BB}}^{-1}}$, but it cannot guarantee that BigBank did not share K_{BB}^{-1} with someone else. These assumptions thus create risks. Identifying a protocol's required assumptions enables us to understand the ways in which it might fail other than by being broken.

The way we express required assumptions affects how much we can learn about the ways a protocol might fail. If we are imprecise, we might underestimate the risk of using the protocol. Instead of Assumption Set 4.2.2, we might have described Alice's assumptions thus:

1. For any message M and public key K, if I receive $M\{\!|M|\!\}_{K^{-1}}$ then someone with K^{-1} sent it.

2. For any principal P and public key K, if Auth says that only P has K^{-1}, then Auth believes this.

3. For any principal P and public key K, if Auth believes that only P has K^{-1}, then it is true.

This is quite concise and clearly shows the assumption of honesty and the assumption of competence. Compared to Assumption Set 4.2.2, though, it leaves some important questions unanswered, such as: How (i.e., with what message) does Auth say that only P has K^{-1}? What name does Auth use for P? How will a message be signed? Even without knowing it, by participating in the protocol Alice will effectively assume *some* answers to these questions, and she may be wrong.

Consequences of Failure

If Alice's assumptions are false, then BigBank might not have sent $M\{\!|M|\!\}_{K_{\mathrm{BB}}^{-1}}$. Alice could end up sharing her banking credentials with a criminal, which is clearly bad. But PKI protocols like this one can be used in other, less risky ways. Suppose that, when Alice communicates with BigBank for the first time (using this protocol), all of her assumptions are true, and when finished she retains BigBank's certificate, perhaps so that she can easily encrypt messages to BigBank with BigBank's public key. Later, Auth is somehow compromised and makes $\mathrm{sigcert}(N_{\mathrm{BB}}, K_{\mathrm{P}}, K_{\mathrm{Auth}})$, where K_{P} is the public key of an attacker. If Alice still relies on Auth to tell her what BigBank's public key is, then she is in trouble, but fortunately she just continues to use K_{BB}. In her communications with BigBank, she no longer needs to trust Auth (or at

least she does not need to believe that Auth is trustworthy *at this time*), because now her only assumption is that BigBank's public key is K_{BB}; she just needs to trust BigBank to protect K_{BB}^{-1}.

4.3 OpenID

4.3.1 For Adult Verification

OpenID ([41]) is an authentication scheme of the kind advocated by the While House's NSTIC (discussed in the beginning of Chapter 1). The main function is to enable a user to use the same name to log into multiple (unaffiliated) websites. One principal, the "identity provider," is responsible for ensuring that only that user can use his or her ID by, for example, requiring the user to ender a password. Using the OpenID protocol, the website that the user wants to log into (the "relying party") interacts with the user and the identity provider so that the identity provider authenticates the user and then tells the relying party that the user was authenticated, without revealing the user's credentials (usually a password) to the relying party.

OpenID can also be used to exchange information about a user when the protocol is combined with an extension called 'Attribute Exchange' ([42]). In this case, the identity provider happens to be an authority (at least in someone's opinion) about some aspects (e.g., age) of its clients, and the relying party happens to need to know about just such aspects. The user can have the identity provider assert to the relying party that, for example, he or she has a certain age or belongs to a certain security group, with or without revealing his or her ID to the relying party. We will consider an example of this way of using OpenID.

Forget what we have said about Alice, BigBank, and Auth. Now suppose that Alice is a potential customer of BigBank, that BigBank requires a potential customer to be an adult in order to open an account, and that Auth is an identity provider that is willing to assert that Alice, who has an ID with Auth, is an adult. Let N_{A} be Alice's ID. With Attribute Exchange, the relying party tells the identity provider what attribute it wants the value of, and then the identity provider responds with that value. Suppose there is an attribute — we will call it 'the adulthood attribute' — that indicates whether the subject is an adult; its possible values are 'true' and 'false'.

Suppose that the run shown in Figure 4.3 occurs. OpenID with Attribute Exchange is much more complex than the previous two protocols, and so I will describe each step:

1. Alice sends BigBank a message ($M_{\mathrm{new\_acct}}$) that says that she would like to open an account, along with N_{A}, which is her OpenID ID. She might do this by visiting a particular

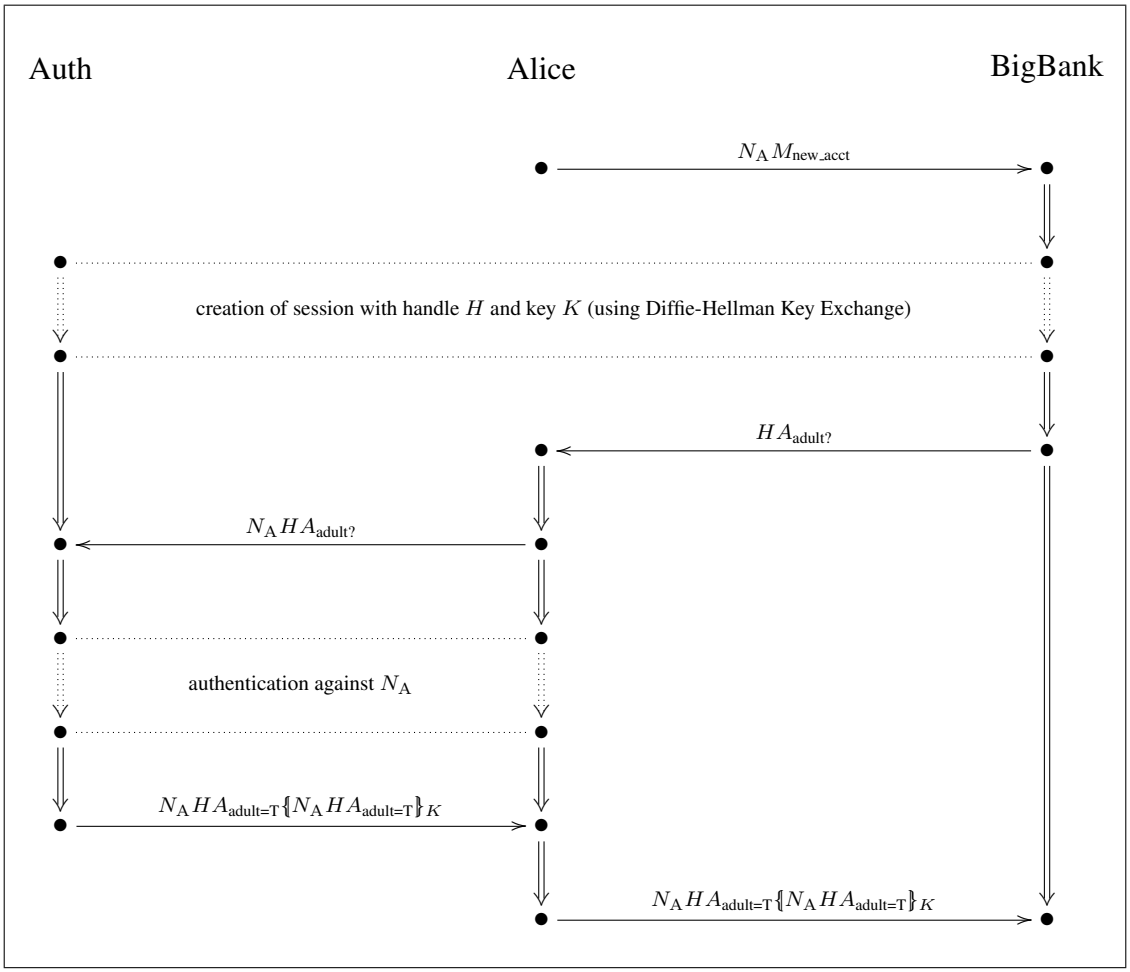

Figure 4.3: Using OpenID for Adulthood Verification

URI (e.g., '/createAccount') or by submitting an HTML form; in any case 'M_{new_acct}' denotes the part of the message that indicates Alice's intention to BigBank.

2. BigBank and Auth use Diffie-Hellman Key Exchange ([43]) to agree on a shared key K and on an "association handle" H, which they will use to recognize messages belonging to this conversation.

3. BigBank sends Alice $HA_{adult?}$, which she is supposed to forward to Auth. $A_{adult?}$ is a string that requests (in accordance with the Attribute Exchange specification) the value of the adulthood attribute. (Actually, BigBank includes N_A in the message as a convenience to Alice, but I left this out since it is not important for this analysis.)

4. Alice sends $N_A HA_{adult?}$ to Auth.

5. Auth somehow authenticates Alice against N_{A}.

6. Auth sends Alice the message $N_{\mathrm{A}} H A_{\text{adult=T}} \{\!|N_{\mathrm{A}} H A_{\text{adult=T}}|\!\}_K$, which she is supposed to forward to BigBank. $A_{\text{adult=T}}$ is a string that says (again, in accordance with the Attribute Exchange specification) that the adulthood attribute's value is 'true'.

7. Alice forwards $N_{\mathrm{A}} H A_{\text{adult=T}} \{\!|N_{\mathrm{A}} H A_{\text{adult=T}}|\!\}_K$ to BigBank.

In the previous sections, we considered Alice's point of view. Here, we will consider BigBank's. What is the goal of this run with regard to BigBank? That is, what is it supposed to convince BigBank of? Consider this possible formulation of the goal belief:

Alice is an adult.

In a way, this is correct; indeed, if we were to ask Alice, she would say something like 'The goal is to convince the bank that I am an adult'. But look at the string that Auth signed and sent (indirectly) to BigBank — $N_{\mathrm{A}} H A_{\text{adult=T}}$: N_{A} is not necessarily 'Alice', and BigBank does not necessarily know whom N_{A} belongs to. It is true that, in Goal Belief 4.2.1, we used the name 'BigBank' in describing Alice's beliefs, but in those examples Alice starts out knowing BigBank in a way that BigBank, in the present example, may not know Alice. We want this protocol to work even if BigBank initially knows nothing about Alice, but in such a case it makes no sense to describe the belief arising from $N_{\mathrm{A}} H A_{\text{adult=T}}$ with a sentence containing the name 'Alice'. Therefore, a better formulation of the goal belief is this:

Goal Belief 4.3.1 (For BigBank)**.** *The principal whom Auth calls N_{A} is an adult.*

Now we can consider BigBank's assumptions:

Assumption Set 4.3.1 (For BigBank)**.**

1. *For any string H, OpenID ID N, and symmetric key K, if I have established a session with Auth whose handle is H and whose key is K, then if I receive $NHA_{adult=T}\{\!|NHA_{adult=T}|\!\}_K$ then Auth believes that the principal whom it calls N is an adult.*

2. *If Auth believes that some principal is an adult, then that principal is an adult.*

As in Assumption Set 4.2.2, we have one assumption about Auth's honesty and another about Auth's competence.

The danger for BigBank of using this protocol is that the principal whom Auth is currently calling N is not an adult. The risk depends on the consequences of BigBank opening an account for a child; they are probably not catastrophic. But in the next subsection we will look at what happens if BigBank uses OpenID for another purpose.

4.3.2 For Authentication

We will plausibly extend this example by supposing that customers can log into BigBank's website using their OpenID IDs — after all, BigBank was already using these IDs to verify adulthood. The protocol (Figure 4.4) is almost the same as before.

Suppose Alice has established an account at BigBank and now uses her OpenID ID, N_{A}, to log into BigBank's website. Suppose the run shown in Figure 4.4 occurs and afterward Alice can access her account. Again, we will look at the example from BigBank's point of view. The goal belief must induce BigBank to let Alice access her account, and so to find the goal belief we must consider how Alice will access her account. Notice that, at the beginning of the run, BigBank sends Alice the string N_{SID}: this is a tag that they will both use to recognize messages in this session (perhaps by sending it back and forth in an HTTP cookie). Thus Alice will access her account using N_{SID} as a session ID. Immediately after the first message is sent, BigBank does not know whom it is talking to, but it does have a name for this principal: N_{SID}. The goal belief must attribute some property ϕ to the principal called N_{SID} that is sufficient (according to BigBank's security policy) to be allowed access to Alice's account. It therefore must have the following form:

$$\text{For any principal } x\text{, if I call } x\ N_{\mathrm{SID}} \text{ then } \phi(x).$$

At first it seems obvious that ϕ is the property of *being Alice*, but what does it mean to say 'BigBank believes that the principal called N_{SID} is Alice' when (as we discussed above) BigBank probably does not really *know* Alice? Since Alice's account is a bank account, it must include a set D of data — such as Alice's full name, address, and even Social Security number — that we would consider to be "identifying" in that there is a conventional and objective way of interpreting it in order to find significant artifacts of a unique person, such as that very person, that person's house or family, or more identifying data about that person. So, rather than claiming that BigBank requires the user to "be Alice," we can claim that it requires the user to

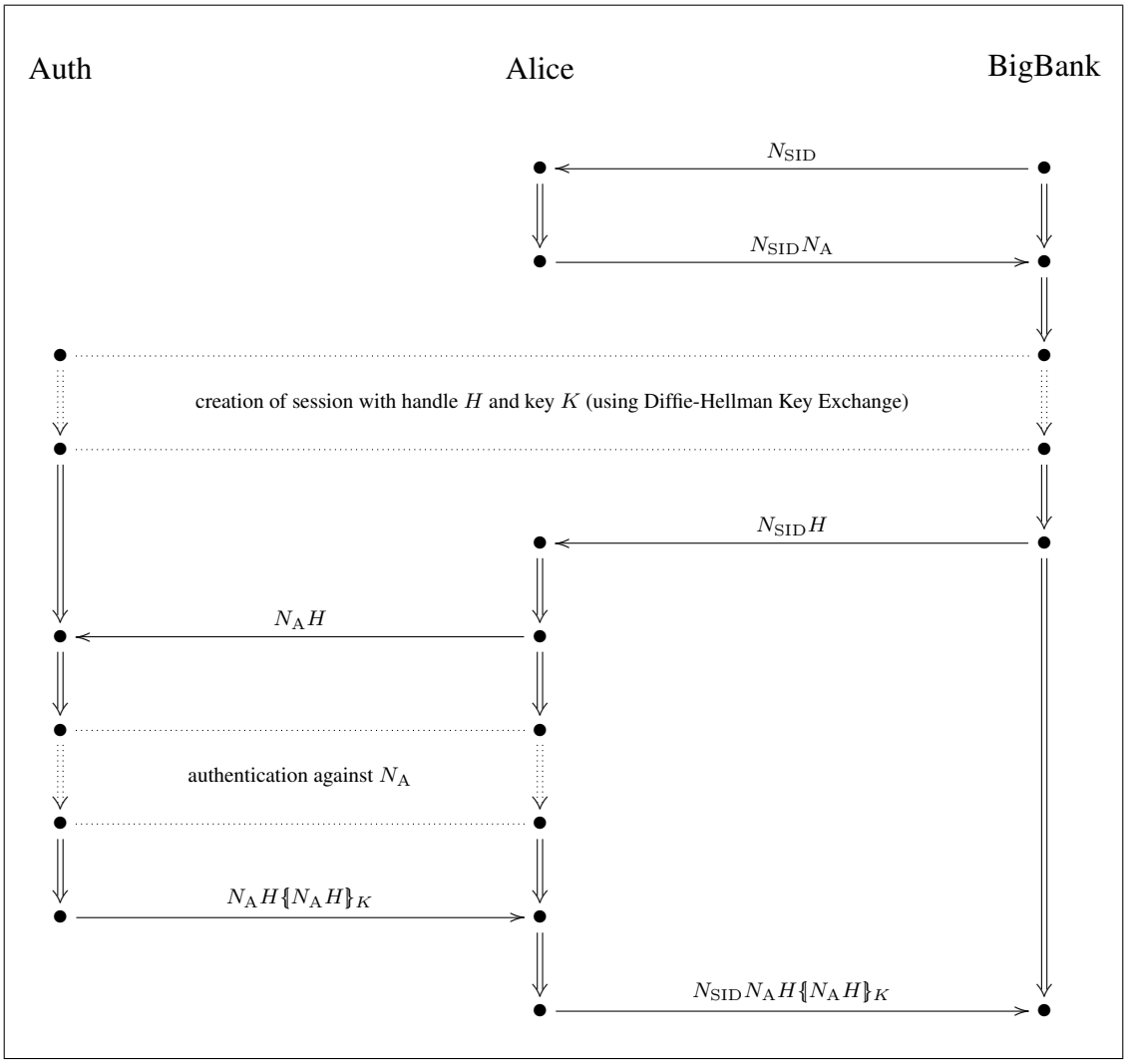

Figure 4.4: Using OpenID for Authentication

be the person thus identified by D — in other words, to be the person whom BigBank calls D. Perhaps now we can state the goal belief:

For any principal x, if I call x N_{SID} then I call x D.

There is something wrong this this formulation, however. Notice that we are using 'call' with two different senses. To be "called N_{SID}" by BigBank is not to be described by N_{SID}, since N_{SID} does not describe anything — it is just an arbitrary value. Rather, it is to be able to send messages containing N_{SID} to BigBank, and whether a particular principal has this ability

depends on things like whether N_{SID} was encrypted when it was sent, whether Alice has shared N_{SID}, and so on, and it is possible to imagine anyone having this ability, just as it is possible to imagine that our solar system has only five planets. In other words, it is contingent. On the other hand, to be "called D" by BigBank is, effectively, to be Alice, because a world where BigBank calls someone else D makes as much sense as a world where BigBank calls an integer other than 9 'the successor of 8'. Clearly N_{SID} and D have different roles in this interaction — they are different kinds of names corresponding to different meanings of 'call'. It turns out that David Kaplan has also discovered these (and other) kinds of names. In Kaplan's terms (cf. [29]), D is a "vivid name" that "denotes" Alice, whereas N_{SID} is such that, if BigBank calls Alice N_{SID}, then N_{SID} is "*of*" Alice, but it never "denotes" Alice because it has no "descriptive content." In just a few steps, our analysis of authentication has wondered into the territory of philosophy. Going deeper than this would lead us out of the scope of this thesis, and so we will just acknowledge the two meanings of 'call' and distinguish between them with subscripts: 'call$_1$' will indicate that the name in question is non-vivid, while 'call$_2$' will indicate that it is vivid. We thus have the following:

Goal Belief 4.3.2 (For BigBank)**.** *For any principal x, if I call$_1$ x N_{SID} then I call$_2$ x D.*

(For more on these philosophical issues, cf. [25], [28], and especially [29].)

So how does OpenID achieve this goal? When Alice's account was created, BigBank associated N_{A} with it, effectively creating the policy that whomever Auth calls N_{A} is allowed to access the account, and so BigBank must believe that whomever Auth calls$_1$ N_{A} BigBank calls$_2$ D. Which sense of 'calls' holds between Auth and N_{A}? OpenID IDs are not very descriptive (and certainly not as descriptive as identifiers like D), and so it seems best to choose 'calls$_1$'. Thus all the protocol needs to do to convince BigBank of the goal belief is to convince BigBank that whomever it calls$_1$ N_{SID} Auth calls$_1$ N_{A}. We therefore posit that BigBank makes the following assumptions:

Assumption Set 4.3.2 (For BigBank)**.**

1. *For any string H, OpenID ID N, session ID N_{SID}, and session key K, if I have established a session with Auth whose handle is H and whose key is K and if I have sent $N_{\text{SID}}H$, then if I receive $N_{\text{SID}}N_{\text{A}}H\{\!|N_{\text{A}}H|\!\}_K$ then whomever I call$_1$ N_{SID} Auth calls$_1$ N_{A}.*
2. *For any principal x, if Auth calls$_1$ x N_{A} then I call$_2$ x D.*

If the distinction indicated with the subscripts on 'calls' seems too academic, try formulating Goal Belief 4.3.2 with only one sense of 'calls'. We could use only 'call$_1$', resulting in this:

$$\text{For any principal } x, \text{ if I call}_1\ x\ N_{\text{SID}} \text{ then I call}_1\ x\ D. \tag{4.1}$$

Now, imagine that, instead of Alice, an attacker plays her role, gets the session ID N_{SID} from BigBank, and sends $N_{\text{SID}} N_{\text{A}}$ to BigBank; and then Auth messes up for some reason and calls$_1$ the penetrator N_{A}, so that BigBank allows the penetrator to access Alice's account. Does this make (4.1) false? BigBank is certainly calling$_1$ the penetrator D, and thus the consequent is true; therefore BigBank cannot be accused of having a false belief. (4.1) thus fails to capture BigBank's security policy. Alternatively, we could use just 'call$_2$' to formulate the goal belief, but then the antecedent would never be true because names like N_{SID} (which is just an arbitrary value) have no meaning. So we see that we must make this distinction between kinds of names in order to understand BigBank's goal belief.

Consequences of Failure

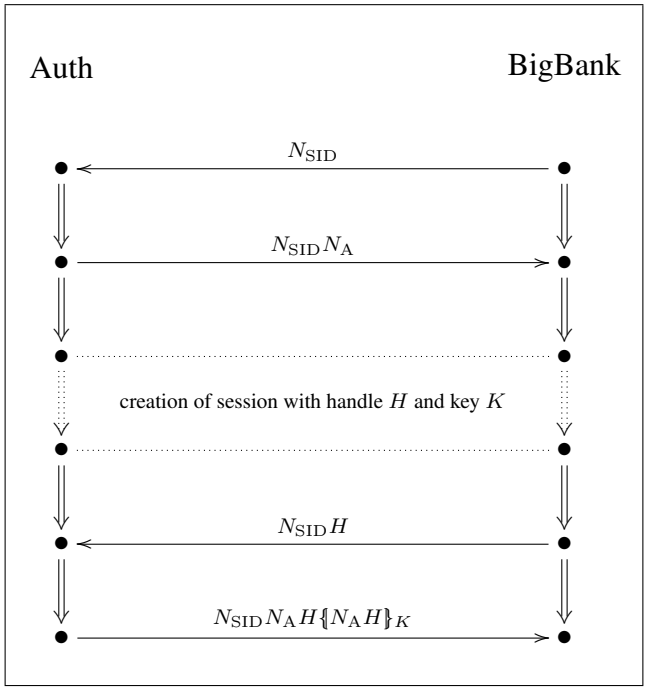

Figure 4.5: Using OpenID for Authentication, with Dishonest Auth

There are many ways in which these assumptions could be false, and many of them can be found with a deeper discussion of the relations denoted by 'calls$_1$' and 'calls$_2$' (along the lines

of [29]). Here, I will just mention that, obviously, one general way is that Auth calls$_1$ someone other than Alice N_{A}. To be concrete, suppose Auth calls$_1$ *itself* N_{A}. The run is now as shown in Figure 4.5.

This is similar to a failure of PKI-based protocols in which the authority signs a bad certificate. But the interesting thing is how the consequences in the present case differ from those in the PKI case. The potential damage caused by Auth's mischief in the PKI case is serious, but it is at least limited by the fact that it does not provide the attacker access to data encrypted with the legitimate public key and it does not hurt Alice if she always trusts the legitimate public key instead of relying on Auth. In the present case, Auth's mischief results in Auth having complete access to Alice's account; BigBank and Alice thus utterly depend on Auth's honesty and competence.

4.4 Anonymous Adulthood Verification

Finally, we will consider a security protocol called 'AdultVerify', for which it is almost impossible to ignore names when formulating the required assumptions. I invented this protocol to use as an example in this thesis. It has not been rigorously reviewed by anyone; although I think that it is secure, it may be broken and therefore should not be used. However, I believe that, if it is broken, any changes needed to fix it would not falsify the analysis presented in this section.

AdultVerify's purpose is very similar to that of the first OpenID example: to convince a server that a client is an adult using an assertion from an authority. The difference is that the client does not want the server to know who he or she is, and does not want the authority to know that he or she is using the server's service. This "anonymity requirement" cannot be satisfied with OpenID.

Applying AdultVerify to Alice, BigBank, and Auth results in the run shown in Figure 4.6. Again, BigBank allows only adults to create accounts. The goal is for Alice to establish an account with BigBank, under a username (N_{X}) provided by BigBank. N_{X} might be sensitive, and so it must not be revealed to Auth; therefore BigBank must encrypt it before sending it to Alice. At the beginning of the run, Alice and BigBank establish a temporary encryption session with session ID N_{SID} and symmetric key K_{dh}, which is created through Diffie-Hellman Key Exchange. (If you are not familiar with Diffie-Hellman Key Exchange, you just need to know that in Diffie-Hellman two principals send each other (unencrypted) integers from which they can compute a symmetric key; the "magic" is that only those two principals can compute that

key, even if a penetrator intercepted both of the exchanged integers.)

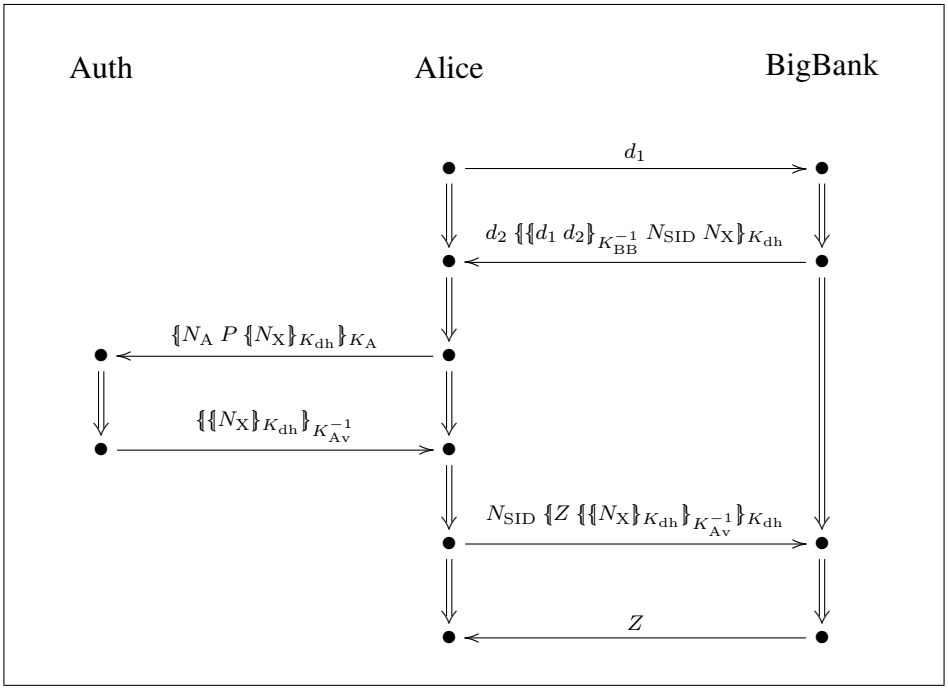

Figure 4.6: Using AdultVerify.

The run is as follows:

1. As the first step in Diffie-Hellman Key Exchange, Alice picks and sends the integer d_1.

2. BigBank sends $d_2\ \{\{\{d_1\ d_2\}_{K_{\mathrm{BB}}^{-1}}\ N_{\mathrm{SID}}\ N_{\mathrm{X}}\}_{K_{\mathrm{dh}}}$. d_2 is BigBank's half of Diffie-Hellman, and K_{dh} is the symmetric key made from d_1 and d_2. In the encrypted part of the message, $d_1 d_2$ is signed with BigBank's private key K_{BB}^{-1}, and N_{SID} and N_{X} are nonces that BigBank generated. N_{SID} will be used as the session ID, and N_{X} will be Alice's username.

3. Alice now must prove that she is an adult to BigBank. She sends $\{\{N_{\mathrm{A}}\ P\ \{\{N_{\mathrm{X}}\}_{K_{\mathrm{dh}}}\}_{K_{\mathrm{A}}}$ to Auth. K_{A} is Auth's public key for encryption; N_{A} is the ID Alice uses with Auth, and P is her password. If Auth believes that she is an adult, it will sign $\{\{N_{\mathrm{X}}\}_{K_{\mathrm{dh}}}$ with the private key K_{Av}^{-1}, which it uses only for this purpose.

4. Auth verifies the ID N_{A} and the password P, and because it believes that Alice is an adult, it sends her $\{\{\{N_{\mathrm{X}}\}_{K_{\mathrm{dh}}}\}_{K_{\mathrm{Av}}^{-1}}$.

5. Alice forwards $\{\!|\{\!|N_{\mathrm{X}}|\!\}_{K_{\mathrm{dh}}}|\!\}_{K_{\mathrm{Av}}^{-1}}$ to BigBank by sending N_{SID} $\{\!|Z$ $\{\!|\{\!|N_{\mathrm{X}}|\!\}_{K_{\mathrm{dh}}}|\!\}_{K_{\mathrm{Av}}^{-1}}|\!\}_{K_{\mathrm{dh}}}$. Z is a nonce that Alice chooses at this point.

6. BigBank acknowledges Alice's message by sending back the nonce Z.

We will consider this run from BigBank's point of view. As in the first OpenID example, BigBank's goal belief is something like 'The principal whom I call X is an adult'. At the end of the run, BigBank has two names for the principal with whom it is communicating: N_{SID} and N_{X}. As in the first OpenID example, N_{SID} is just a temporary name, and we want the goal belief to refer to a permanent name, like N_{X}. N_{X} is not a descriptive name, and so 'call$_1$' is more appropriate than 'call$_2$'. We thus get the following formulation:

Goal Belief 4.4.1 (For BigBank)**.** *The principal whom I call$_1$ N_{X} is an adult.*

One of the required assumptions is that Auth is right when it thinks that someone is an adult. The other one is an interpretation of a class of messages — namely, messages like $\{\!|\{\!|N_{\mathrm{X}}|\!\}_{K_{\mathrm{dh}}}|\!\}_{K_{\mathrm{Av}}^{-1}}$ — as Auth's way of saying that it thinks someone in particular is an adult. In the first OpenID example, BigBank knows which name Auth will use for the client — namely, N_{A} — and so it is accurate for BigBank to attribute the belief 'The principal whom I call N_{A} is an adult' to Auth, but in the current example BigBank does not know N_{A}. Therefore the assumption should not say anything about the name that Auth uses to refer to the client.

Assumption Set 4.4.1 (For BigBank)**.**

1. *If during a run of this protocol in which N is the username and K is the session key I receive $\{\!|\{\!|N|\!\}_K|\!\}_{K_{\mathrm{Av}}^{-1}}$, then the principal whom I call$_1$ N is believed by Auth to be an adult.*

2. *If Auth believes that some principal is an adult, then that principal is an adult.*

In the first assumption, it would be misleading to attribute the belief 'The principal whom BigBank calls N is an adult' to Auth, because the protocol is designed so that Auth does not know N and is even unaware of BigBank's involvement.

4.5 Discussion

We have seen that even simple protocols like the second message-signature protocol can require non-trivial assumptions. In general, these assumptions fall into the following (overlapping) categories:

1. Assumptions about the meaning of a message.
2. Assumptions about the use of a name or class of names.
3. Assumptions asserting the honesty of a principal.
4. Assumptions asserting the competence of a principal.

Assumptions of the last two categories are usually more obvious than assumptions of the first two. To fully understand the risks of using a protocol, our formulation of the required assumptions should be precise enough to include those in the first two categories. But, as we have seen, doing so can take us into tricky philosophical territory. Surprisingly, there is a body of work in the philosophy of language going back more than a hundred years that is quite relevant to computer security protocols. Of course, from the beginning computer science has had a number of significant connections to philosophy, and so perhaps we should not be surprised by yet another.

Most of the formulations of the goal beliefs and required assumptions in this chapter have resulted in convoluted sentences, and the cause has been our determination to explicitly state assumptions about the use of names. It is easier to express these beliefs and assumptions using a formal language. We could then come up with some formal inference rules and use them to show that the goal beliefs follow from the assumptions. But proper use of a formal logic requires, in Paul Syverson's words, "an independently motivated semantics" ([32, p. 165]), which in turn requires a model of our problem domain. In the next chapter, we take a step toward developing a model of security protocols like those considered here that includes principals' beliefs and their use of names, and we show how this model could be used as the basis of a formal logic.

CHAPTER 5:
Toward a Mathematical Model

5.1 A Running Example: SimpleAdultVerify

Suppose we have a very simple protocol called 'SimpleAdultVerify' with which some authority can reliably send unforgeable assertions about people's ages. Specifically, SimpleAdultVerify has two roles, Auth and Client, and a run consists of Auth sending Client a message saying either that some person is an adult or that Auth does not know whether some person is an adult. (Auth does not keep track of who is *not* an adult.) Client would be an online application that needs to restrict access to certain age-groups. An assertion from Auth must be about some person, and it must refer to that person by some name that is meaningful to both Auth and Client. We will assume that SimpleAdultVerify is used in an environment in which every principal has at most one name and everybody knows to whom these names refer. For simplicity, we will also assume that Auth always serves only one Client, and so it does not need to keep track of who sent what query.

In this chapter, we will attempt to build a model of an application of SimpleAdultVerify to a world containing four principals — a, c, j, and k. a plays the role Auth, c plays Client, and j and k do not play any role. When we make an important decision about the model, we will express it as a "postulate." Our first postulate expresses the goal of this protocol:

Postulate 5.1 (Successful Runs of SimpleAdultVerify)**.** *A run of SimpleAdultVerify is successful if and only if there exists a name N such that exactly one of the following is true:*

- *Throughout the run a believes that the principal named N is an adult, and at the end so does c.*
- *Throughout the run, a does not know whether the principal named N is an adult and c does not change its belief about it.*

The questions that we want our model to answer are the following: What assumptions does c have to make for a run to be successful? How do principals' beliefs change during a run?

5.2 Modeling Actions

In this section, we will build a model, S_{SAV}, of all the ways the principals can interact in our application of SimpleAdultVerify — that is, all the possible *runs* of the protocol application.

S_{SAV} is an example of a kind of model that we will call 'a protocol system'. Following the multi-agent system approach (cf. Section 2.1), a protocol system models a protocol application as a system of principals with finite local states. Runs are sequences of global states, and each transition corresponds to one or more exchanges of messages prescribed by the protocol. Our current task consists of two steps: defining the structures of the principals' local states, and defining the transition function. We will depart from the standard way of defining multi-agent systems by requiring that runs be finite; runs of security protocols usually have a definite ending, and it is often useful to refer to the system's state at this ending.

Definition 1. *A* protocol system *is a tuple* $(\mathcal{P}, \mathcal{T}, \mathcal{L}, \mathcal{R})$ *where*

- $\mathcal{P}$ *is the set of principals in the system.*
- $\mathcal{T}$ *is the set of texts.*
- *For each* $p \in \mathcal{P}$*, there exists a set* $L_p \in \mathcal{L}$*, which is the set of all possible local states of* p*.*
- $\mathcal{R}$ *is a set of runs.*

Let $S_{SAV} = (\mathcal{P}, \mathcal{T}, \mathcal{L}, \mathcal{R})$. In the rest of this section, we will specify the details of S_{SAV}. We can start with two obvious ones:

Postulate 5.2 (S_{SAV}'s Principals)**.** $\mathcal{P} = \{a, c, j, k\}$

Postulate 5.3 (Local-State Sets)**.** $\mathcal{L} = \{L_a, L_c, L_j, L_k\}$

Now we consider the principals' local states. For most protocol applications, there is no single right way to design the local states and the transition function; the usefulness of a given design will depend on the interpretation the analyst gives to the principals' local states. While there are often many alternative designs, we will construct S_{SAV} in accordance with the following principle: *a principal's local state only represents previous or potential behavior, in accordance with the protocol, of that principal*. In particular, a local state must not be intended to indicate

how data is stored by the principal or the principal's beliefs. When defining the structure of a local state, it is a good idea to provide a behavioral interpretation for each element of the structure. Since a principal's behavior or history will always involve messages that are sent or received, their local states will include sets of or relations over *texts*. The possible texts are specified in the protocol system with $\mathcal{T}$.

We start with the local states of a. Clearly, a's local states must contain a set $adultNames$ of names of principals whom a would affirm to be adults (notice that I did not write 'whom a believes to be adults'). $adultNames$ will not change during a run. We also add a set $queries$ of names contained in queries sent by c that a has not yet responded to. Since we are assuming that c is the only principal that sends queries to a, we do not need $queries$ to keep track of the sender. So, we make the following postulate:

Postulate 5.4 (a's Local States)**.** *Every local state in L_a consists of the following elements:*

- $adultNames \subseteq \mathcal{T}$*: the set of names of the principals whom a would assert to be adults.*
- $queries \subseteq \mathcal{T}$*: the set of names of the principals about whom c has queried but a has not yet sent an assertion.*

(Notice that we give a behavioral interpretation for each element.)

We now specify the structures of the local states of the other principals in a similar manner. c's behavior is to send a query and then receive the response:

Postulate 5.5 (c's Local States)**.** *Every local state in L_c consists of the following elements:*

- $queriesToSend \subseteq \mathcal{T}$*: the set of names about which c will (in the current or another run) send a query.*
- $assertedAdultNames \subseteq \mathcal{T}$*: the set of names of principals whom a has asserted to be adults.*

j and k have no prescribed behavior and therefore always have empty local states:

Postulate 5.6 (j's and k's Local States)**.** $L_j = L_k = \{()\}$

Having specified the structures of the principals' local states, we now specify the set $\mathcal{R}$ of runs of the system. Runs consist of global states, and a global state is a tuple $(s_a, s_c, (), ())$ where $s_a \in L_a$ and $s_c \in L_c$ (the last two elements are j's and k's local states, which are always empty). Each run consists of two steps: (1) c sends a query to a containing some name, and (2) a responds either that the named principal is an adult or that a does not know whether the named principal is an adult. Just as for local states, there are various reasonable ways to specify $\mathcal{R}$ that differ in, for example, the allowed initial states or the allowed lengths of the runs. The following is the least restrictive way that works with the postulates above and captures the principals' behavior:

Postulate 5.7 (Runs)**.** *$\mathcal{R}$ consists of all runs r of length 3 that satisfy the following:*

- *There exists a text X in $\mathcal{T}$ such that $r_c(1).queriesToSend = r_c(0).queriesToSend \setminus \{X\}$ and $r_a(1).queries = r_a(0).queries \cup \{X\}$; $r(0)$ and $r(1)$ are the same in other respects.*
- *There exists a text X in $\mathcal{T}$ such that $r_a(2).queries = r_a(1).queries \setminus \{X\}$, $X \in r_a(1).adultNames$ (and $r_a(1).adultNames = r_a(2).adultNames$), and $r_c(2).assertedAdultNames = r_c(1).assertedAdultNames \cup \{X\}$; $r(1)$ and $r(2)$ are the same in other respects.*

We have now completely specified S_{SAV}. Figure 5.1 shows a run of S_{SAV}.

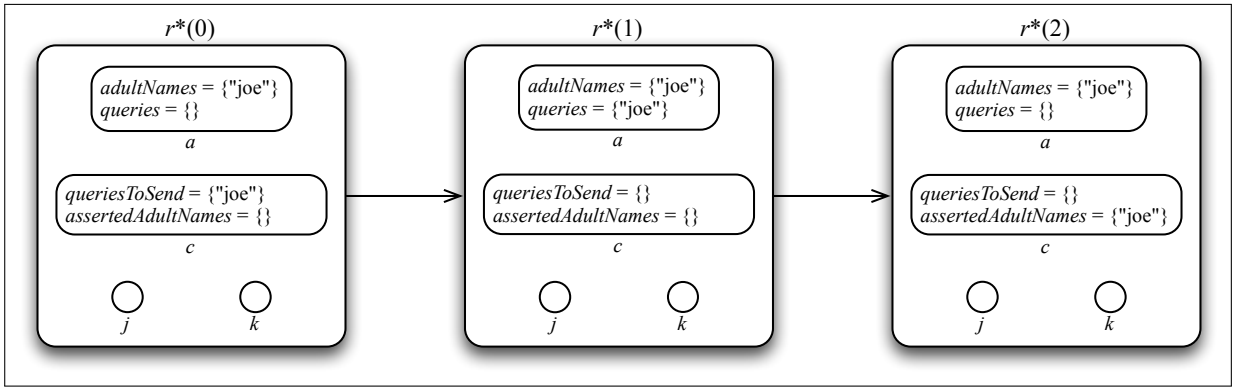

Figure 5.1: A run of S_{SAV} in which c asks a whether the principal called 'joe' is an adult and a replies that he is.

5.2.1 A Formal Language

Although a model like S_{SAV} is a useful description of a protocol application, it is often easier to use a formal language whose semantics is defined in terms of such models. We could adapt the quantified modal language developed by Adam Grove ([10]; cf. Section 3.6) for this purpose. Consider r^* in Figure 5.1 to see how we might do this. Suppose our language has a predicate-symbol for each element of a local state — for example, the predicate-symbol '$\mathrm{AdultName}$' corresponds to the set $adultNames$ in a's local state. Just as Grove's logic is multi-sorted, our language has two sorts: principals and texts. $\mathcal{P}$ is the domain of sort principal, and $\mathcal{T}$ is the domain of sort text. Constants of sort text are made with quotation — for example, '`` `joe´ '' is a constant that denotes the text 'joe'. As with Grove's logic, all text-sort variables are uppercase, and all principal-sort variables and constants are lowercase. Formulas in the language are satisfied by points, but otherwise satisfaction ('$\models$') is defined in the manner standard for quantified modal languages (cf. Section 2.3), except that for now we ignore formulas with modal operators. We get the following:

$$(r^*, 0) \models \text{`}\,\mathrm{AdultName}(\text{`}\,\texttt{joe}\,\text{´})\text{'}$$
$$(r^*, 0) \models \text{`}\,\mathrm{QueryToSend}(\text{`}\,\texttt{joe}\,\text{´})\text{'}$$
$$(r^*, 2) \models \text{`}\,\mathrm{AssertedAdultName}(\text{`}\,\texttt{joe}\,\text{´})\text{'}$$
$$(r^*, 2) \models \text{`}\neg\exists X\ \mathrm{QueryToSend}(X)\text{'}$$

5.2.2 Limits of the Model

Notice that, although the interpretations of the local states in r^* capture all the important behavior, they still leave a lot out. For example, they do not tell us whom 'joe' denotes or whether that person is actually an adult. They also do not tell us anything about the beliefs of any of the principals. In the next section, we will enhance our model so that it represents the principals' beliefs and aspects of the world that are outside of the protocol but nevertheless relevant (e.g., adulthood).

5.3 Modeling Beliefs

The concept of 'protocol system' presented in the previous section enables us to describe a protocol in terms of the actions of principals without assuming too much about how it is implemented. It views principals only as things that can compute and exchange messages. It does not, by itself, enable us to understand how a protocol achieves its goals — that is, how princi-

pals come to possess their goal beliefs. There are in fact two related problems with S_{SAV} (and protocol systems in general), which I mentioned at the end of the last section: (1) it leaves out important aspects of the world, such as the set of adults and the assignment of names, and (2) it does not represent principals' beliefs.

In this section, we will use ideas from modal logic and the multi-agent systems approach to confront these problems and create a new model B_{SAV} by modifying S_{SAV}. This will provide us with a good solution for the first problem (Subsection 5.3.1), but leads to some new problems when applied to the second (Subsection 5.3.2). This section ends with a discussion of these new problems.

5.3.1 Modeling Other Aspects of the World

B_{SAV} will be an expansion of S_{SAV}. We can solve the first problem by including relevant external aspects of the world in the local state of a new "pseudo-principal" e that represents the environment in which a run occurs. With e's local states, we can model who the adults are and who has what name. As we did for the other principals, we must define a set L_e of the possible local states of e:

Postulate 5.8 (e's Local States)**.** *Every local state in L_e consists of the following elements:*

- $adults \subseteq \mathcal{P}$*: the set of adults*
- $nameOf : \mathcal{P} \rightarrow \mathcal{T}$*: the assignment of names (a partial function)*

We call the members of L_e 'environment states'. Unlike our specifications of the local states of the other principals, we do not need to provide a behavioral interpretation of the elements of environment states, because the environment-state does not (necessarily) determine behavior. It is just a model of aspects of the world external to the protocol application.

The set $\mathcal{R}$ of runs in B_{SAV} is the same as in S_{SAV} (cf. Postulate 5.7), except that the global states include environment states. Within a run, only a's and c's local states change. An example is shown in Figure 5.2.

5.3.2 Modeling Beliefs

In this section, we add aspects of relational Kripke structures to S_{SAV} in order to model the beliefs of the principals. (For an overview of relational Kripke structures, cf. Section 2.3.)

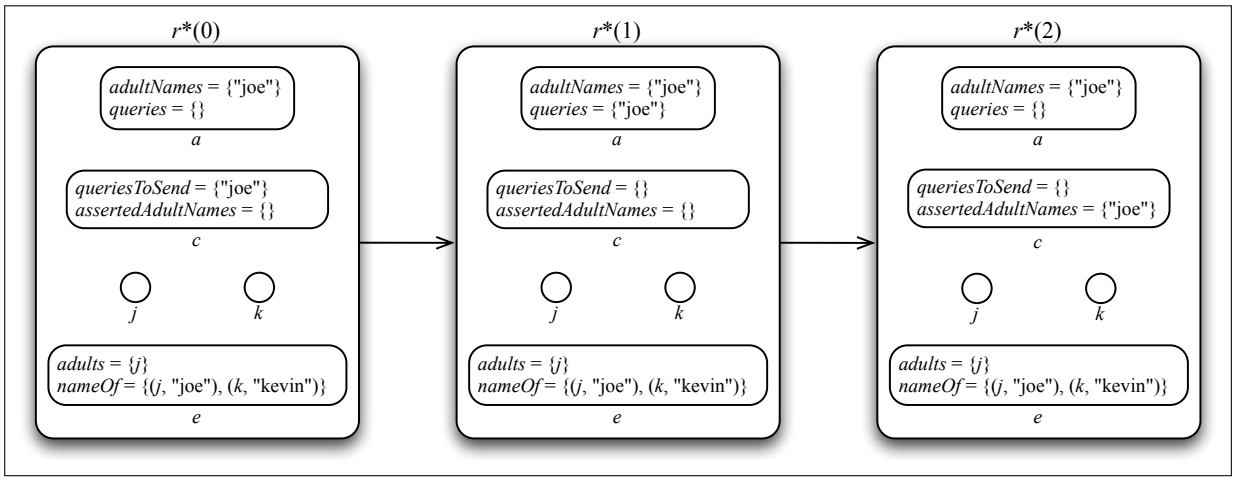

Figure 5.2: A run of B_{SAV} in which c asks a whether the principal called 'joe' is an adult and a replies that he is.

Postulate 5.9. $B_{SAV} = (\mathcal{P}, \mathcal{T}, \mathcal{L}, \mathcal{R}, \mathcal{W}, \mathcal{B})$ *where*

- $\mathcal{P} = \{a, c, j, k\}$
- $\mathcal{T}$ *is the set of texts.*
- $\mathcal{L} = \{L_a, L_c, L_j, L_k, L_e\}$
- $\mathcal{R}$ *the same as in* S_{SAV}*, except that the global states include environment states.*
- $\mathcal{W}$ *is the set of possible worlds.*
- $\mathcal{B} = \{B_a, B_c, B_j, B_k\}$*, and* $B_i \subseteq \mathcal{W} \times \mathcal{W}$ *for each* i *in* $\mathcal{P}$

What exactly should be in $\mathcal{W}$? In the multi-agent system approach (which is concerned with knowledge rather than belief), $\mathcal{W}$ is the set of all points (r, t). The "meaning" of a world (r, t) — that is, the set of propositions that are satisfied by that world — comes from the global state $r(t)$. Even if the set of global states is finite, the set of points is infinite because runs are infinitely long. Because this approach is used to model agents' knowledge rather than beliefs, each accessibility relation K_i is simply defined to include exactly those pairs of points $((r, t), (r', t'))$ where agent i has the same local state in $r(t)$ and $r'(t')$. If we were to do the same thing with B_{SAV}, $\mathcal{W}$ would be $\mathcal{R} \times \{1, 2, 3\}$ (because all runs have a length of 3). For now, we just take $\mathcal{W}$ to be the set of global states — that is, $\mathcal{W} = L_a \times L_c \times L_j \times L_k \times L_e$ — and focus on the belief-relations.

In particular, consider B_c — c's belief-relation — and how it should model c's beliefs during run r^* (Figure 5.2). At the end of the run — that is, in $r^*(2)$ — c should believe that the principal called 'joe' is an adult. This would be modeled by imposing the following condition:

For any w in $\mathcal{W}$ such that $(r^*(2), w) \in B_c$,

for any p in $\mathcal{P}$ such that $w.nameOf(p) =$ 'joe', $p \in w.adults$. (5.1)

Figure 5.3 shows a very small part of what B_c might look like if it satisfied this condition. A very nice feature of this model is that it clearly shows the difference between a name and the principal it denotes, and between a principal's beliefs and the real world. This model highlights the fact that principals know what messages they sent and received, but are still "stuck in their own heads"; to believe something about the rest of the world (e.g., about the environment state) requires an assumption, which could be false.

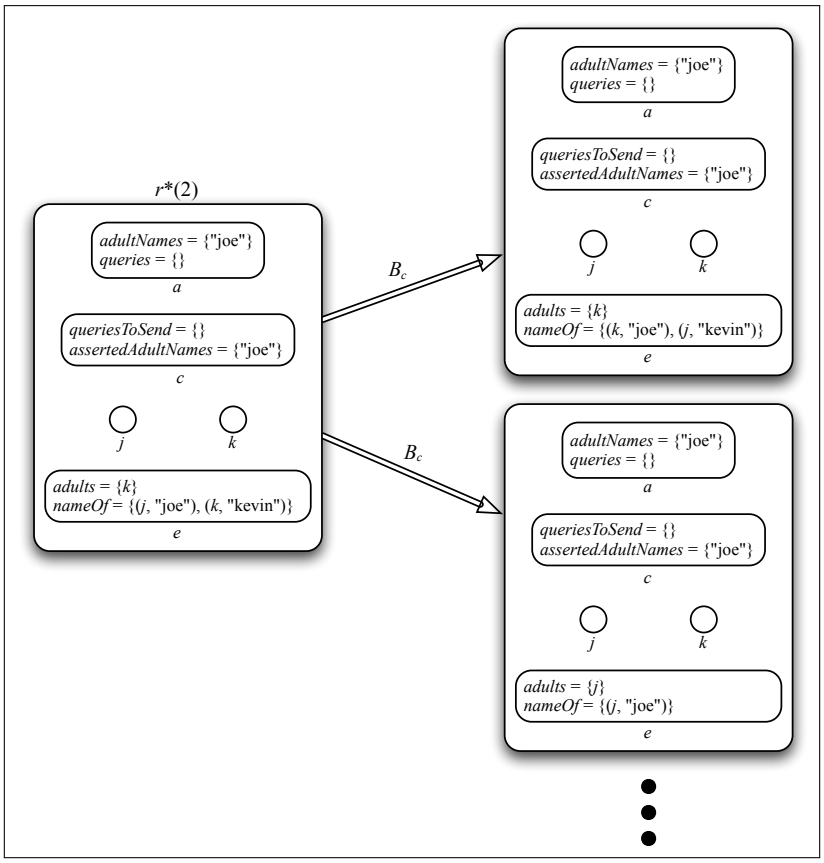

Figure 5.3: A Small Sample of B_c.

It seems that we could develop this model into something like Rangan's (Section 3.1) that captures the assumptions c must make about a's honesty and competence, as well as its knowledge in $r^*(2)$ that a asserted that the principal called 'joe' is an adult. We then would no longer need to explicitly impose (5.1) because the goal belief would logically follow. We could expand the formal language begun in Subsection 5.2.1 into a modal language, add axioms and inference rules appropriate for modal logics of belief, and then construct a formal proof of the goal belief from the assumptions. To represent c's assumption of a's honesty, we would impose this condition:

For any w in $\mathcal{W}$ such that $(r^*(2), w) \in B_c$ and any X in $\mathcal{T}$,
if $X \in w.assertedAdultNames$ then, for any w' in $\mathcal{W}$ such that $(w, w') \in B_a$,
for any p in $\mathcal{P}$ such that $w'.nameOf(p) = X$, $p \in w'.adults$.

This is a mouthful, and so we would express it with the formal language, perhaps thus (cf. the definition of '$\models$' for formulas with modal operators in Section 2.3):

$$\mathbb{B}_c \forall X \Big(\mathrm{AssertedAdultName}(X) \longrightarrow \mathbb{B}_a \forall y \big(\mathrm{NameOf}(y, X) \rightarrow \mathrm{Adult}(y) \big) \Big) \tag{5.2}$$

We would impose a similar condition to represent c's assumption of a's competence. It could be expressed formally with a formula like this:

$$\mathbb{B}_c \forall X \Big(\mathbb{B}_a \forall y \big(\mathrm{NameOf}(y, X) \rightarrow \mathrm{Adult}(y) \big) \longrightarrow \forall y \big(\mathrm{NameOf}(y, X) \rightarrow \mathrm{Adult}(y) \big) \Big) \tag{5.3}$$

Finally, we would require that c knows that it has received a's assertion that the principal called 'joe' is an adult. (Actually, this would probably be a consequence of a general condition that principals know their own states.)

$$\mathbb{B}_c \, \mathrm{AssertedAdultName}(\texttt{`joe´}) \tag{5.4}$$

So, to summarize, we would require that B_c and, incidentally, B_a be such that $r^*(2)$ satisfies (5.2), (5.3), and (5.4). We would next check (perhaps with a formal proof) that this implies that $r^*(2)$ satisfies the condition in (5.1), which would be expressed thus:

$$\mathbb{B}_c \forall y \big(\mathrm{NameOf}(y, \texttt{`joe´}) \rightarrow \mathrm{Adult}(y) \big) \tag{5.5}$$

In other words, if $r^*(2) \vDash$ (5.2), (5.3), and (5.4) then $r^*(2) \vDash$ (5.5). This turns out to be the case, and the formal proof could be done using Modus Ponens, rules for instantiating quantifiers, and a special set of axioms called 'K' that hold for all modal logics with a truth-functional conditional operator (like our '$\rightarrow$') and a possible-worlds semantics. K (adapted to our formal language) is the set of all formulas with the following form:

$$\ulcorner\big(\big(\mathbb{B}_\kappa(\phi \rightarrow \psi) \wedge \mathbb{B}_\kappa\,\phi\big) \longrightarrow \mathbb{B}_\kappa\,\psi\big)\urcorner$$

(ϕ and ψ stand for any formulas and κ for any constant of sort principal.) All formulas in K are valid no matter what the belief-relations are, because their validity comes from the way worlds satisfy formulas whose main operator is '$\rightarrow$'.

Below, we informally prove that, if $r^*(2) \vDash$ (5.2), (5.3), and (5.4) then $r^*(2) \vDash$ (5.5), using the properties of the satisfaction-relation.

Proof. Remember that the "real world" (which is the perspective in which we are doing the proof) is $r^*(2)$. The main operator of the assumptions ((5.2), (5.3), and (5.4)) and the goal ((5.5)) is $\mathbb{B}_c$, and we are thus always talking about c's beliefs, which is the same as talking about all the worlds that c considers possible (in $r^*(2)$). So we switch our perspective from $r^*(2)$ to an arbitrary world w such that $(r^*(2), w) \in B_c$. From the assumptions, we know the following:

$w \vDash$ '$\mathrm{AssertedAdultName}(\grave{}\,\mathtt{joe}\,\acute{})$'
$w \vDash$ '$\forall X\Big(\mathrm{AssertedAdultName}(X) \longrightarrow \mathbb{B}_a\,\forall y\big(\mathrm{NameOf}(y, X) \rightarrow \mathrm{Adult}(y)\big)\Big)$'
$w \vDash$ '$\forall X\Big(\mathbb{B}_a\,\forall y\big(\mathrm{NameOf}(y, X) \rightarrow \mathrm{Adult}(y)\big) \longrightarrow \forall y\big(\mathrm{NameOf}(y, X) \rightarrow \mathrm{Adult}(y)\big)\Big)$'

Therefore,

$w \vDash$ '$\Big(\mathrm{AssertedAdultName}(\grave{}\,\mathtt{joe}\,\acute{}) \longrightarrow \mathbb{B}_a\,\forall y\big(\mathrm{NameOf}(y, \grave{}\,\mathtt{joe}\,\acute{}) \rightarrow \mathrm{Adult}(y)\big)\Big)$'
$w \vDash$ '$\Big(\mathbb{B}_a\,\forall y\big(\mathrm{NameOf}(y, \grave{}\,\mathtt{joe}\,\acute{}) \rightarrow \mathrm{Adult}(y)\big) \longrightarrow \forall y\big(\mathrm{NameOf}(y, \grave{}\,\mathtt{joe}\,\acute{}) \rightarrow \mathrm{Adult}(y)\big)\Big)$'

and so

$w \vDash$ '$\forall y\big(\mathrm{NameOf}(y, \grave{}\,\mathtt{joe}\,\acute{}) \rightarrow \mathrm{Adult}(y)\big)$'

We have now proved that '$\forall y\big(\mathrm{NameOf}(y, \texttt{`joe'}) \rightarrow \mathrm{Adult}(y)\big)$' is satisfied by all points that c considers possible in $r^*(2)$, and this is precisely the condition for $r^*(2)$ to satisfy (5.5). □

5.3.3 Problems with the Model

That was fun, but the proof rests on an uncertain assumption: that such a world as w exists. In fact, it may be that there is no w such that $(r^*(2), w) \in B_c$. This would not mean that $r^*(2)$ does not satisfy (5.5): rather, it would mean that $r^*(2)$ satisfies *all* formulas with the form $\ulcorner \mathbb{B}_c\, \phi \urcorner$! This is because a world w satisfies a formula $\ulcorner \mathbb{B}_c\, \phi \urcorner$ if and only if each world w' such that $(w, w') \in B_c$ satisfies ϕ, and, if there are no such worlds w', then this is trivially the case. In modal logics of belief, this is prevented by ensuring that each belief-relation B_p is *serial* — that is, for each w in $\mathcal{W}$ there exists a w' in $\mathcal{W}$ such that $(w, w') \in B_p$. This means that all formulas with the form $\ulcorner (\mathbb{B}_\kappa\, \phi \rightarrow \neg\, \mathbb{B}_\kappa\, \neg\phi) \urcorner$ are valid, and so they are usually added as axioms (this is the axiom-set called 'D'). But we do not know enough about B_c to determine whether it is serial. Before trying to do proofs, we need to know more about B_c and B_a — in fact, we need to *define* them, rather than just specify constraints on them. It turns out, though, that this is difficult to do.

Problem 1

We have not defined B_c; we have not said what other beliefs c has at $r^*(2)$, nor have we said what c believes at any other world. We have also not defined B_a. One option is to generalize the conditions we imposed on $r^*(2)$ to represent c's assumptions, so that (5.2), (5.3), and (5.4) are satisfied by all worlds. In fact, this makes sense, since c would make these assumptions "*a priori*," independently of the run or the world. But if we take this general approach for all principals' assumptions in every system, we get some unexpected results. For example, it is impossible to model a case where a principal p always believes ϕ (an assumption) but in some world w^* another principal q does not believe that p believes ϕ — formally:

$$w \vDash \ulcorner \mathbb{B}_p\, \phi \urcorner \text{ for any } w \text{ in } \mathcal{W} \tag{5.6a}$$

$$w^* \vDash \ulcorner \neg\, \mathbb{B}_q\, \mathbb{B}_p\, \phi \urcorner \tag{5.6b}$$

Set (5.6) is unsatisfiable because (5.6b) implies that there exists some world w' in $\mathcal{W}$ such that $(w^*, w') \in B_q$ and $w' \nvDash \ulcorner \mathbb{B}_p\, \phi \urcorner$, which contradicts (5.6a). So it seems that we need to choose just a subset of worlds in which principals make their assumptions. The problem is that I know no natural criterion for choosing this subset. In general, a description of a situation that we are

trying to model will only tell us what beliefs hold in some of the worlds, and our modeling technique does not have a general principle for filling out the rest of the belief-relations. For example, in the multi-agent system approach (in which $\mathcal{W}$ is the set of points), the knowledge-relation K_i for any agent i is defined as the set of all pairs $((r,t),(r',t'))$ such that i is in the same local state in $r(t)$ and $r'(t')$; thus, given a set of agents and a set $\mathcal{R}$ of runs, this principle completely defines the knowledge-relations. However, even if we had such a principle for belief-relations, we would have another problem to deal with, which we consider next.

Problem 2

The second problem is that a given definition of $\mathcal{W}$ might restrict which sets of beliefs we can model with the belief-relations. To see how, consider a very simple example. Forget about our protocol models and even relational Kripke structures, and instead suppose that a world consists of just a bit, whose value is either 0 or 1. Suppose our formal language is a propositional modal logic, and in this language 'p' denotes the proposition that the bit's value is 1. There are two principals, a and b. For now, let $\mathcal{W}$ contain just two worlds — one world in which the bit's value is 0, and another in which it is 1. Suppose we want to impose the following conditions on the belief-relations B_a and B_b:

- a always (i.e., in all worlds) knows the bit's value.
- There is a world w^* in $\mathcal{W}$ in which a believes that b believes that the bit's value is 1, but in fact b does not believe this.
- Both belief-relations are serial.

Formally:

$$w \vDash \text{`}\big((p \rightarrow \mathbb{B}_a\, p) \wedge (\neg p \rightarrow \mathbb{B}_a\, \neg p)\big)\text{' for any } w \text{ in } \mathcal{W} \tag{5.7a}$$

$$w^* \vDash \text{`}(\mathbb{B}_a\, \mathbb{B}_b\, p \wedge \neg\, \mathbb{B}_b\, p)\text{'} \tag{5.7b}$$

$$w \vDash \ulcorner(\mathbb{B}_\kappa\, \phi \rightarrow \neg\, \mathbb{B}_\kappa\, \neg\phi)\urcorner \text{ for any } w \text{ in } \mathcal{W} \tag{5.7c}$$

The first two conditions are mundane. The last one — seriality — is important because without it there are worlds for which at least one of the belief-relations is meaningless, in that it causes a principal to believe everything in these worlds.

It turns out that it is impossible to satisfy conditions (5.7) with just the two worlds in $\mathcal{W}$. If we drop seriality ((5.7c)) we can satisfy (5.7a) and (5.7b), but the meanings of the belief-relations then become questionable. The problem is not that these conditions are mutually contradictory: in fact, if we add to $\mathcal{W}$ an additional world in which the bit's value is 1, then all three conditions can be satisfied. One way of doing this is shown in Figure 5.4. In this example, the bit's value happens to be 1 in w^*, and so its value must be 1 in every world that a considers possible at w^*. a has a false belief about b's belief about the bit, and so a cannot consider w^* possible; therefore, we need an extra world in which the bit's value is 1 and a's belief about b's belief is true. In general, it seems that this problem comes from principals having false beliefs about other principals' beliefs, which causes the belief-relations to interfere with each other if there are not enough worlds in $\mathcal{W}$.

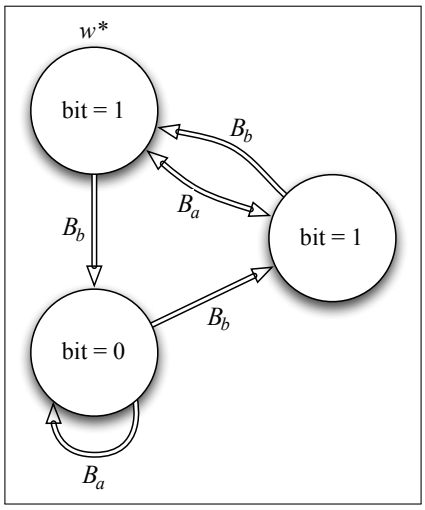

Figure 5.4: One way of satisfying conditions (5.7) with an extra world.

The same thing can happen with B_{SAV}. In Section 5.2 and Subsection 5.3.1 we came up with well-motivated definitions of the local-state sets L_i and the environment-state set L_e, and it seems that defining $\mathcal{W}$ as $L_a \times L_c \times L_j \times L_k \times L_e$ would capture all the interesting possibilities for the protocol application. Nevertheless, if $\mathcal{W}$ is finite (which it is if and only if $\mathcal{T}$ is finite), I would expect that there are still sets of reasonable conditions that cannot be satisfied. This will not be a problem if $\mathcal{W}$ is infinite, but then we get the first problem discussed above.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 6:
Conclusion

This thesis has presented a view of security protocols as devices that enable principals to acquire security-relevant beliefs based on their "technical" knowledge and beliefs (about, e.g., cryptography and computer networks), on their assumptions about the meanings of messages and the names that appear in messages, and on their trust in other principals. We call these beliefs a protocol's "required assumptions."

We found that protocols' required assumptions are often less about principals than about keys and names and, in particular, the way names are used. We then quickly discovered that to understand such assumptions we must make some very subtle distinctions between different ways of naming, and such distinctions are usually made by philosophers rather than computer scientists, and certainly not by the average user who encounters TLS most often when shopping at Amazon. And yet these distinctions can be essential to understanding how a security protocol is supposed to achieve its goals, as we saw with the example of using OpenID for authentication (Subsection 4.3.2).

The complexity of some of the required assumptions calls for a formal modal language that can deal with naming. This, in turn, calls for a meaningful model on which to build a semantics. We took an initial step toward this, but found problems with modeling belief using Kripke structures while requiring that the parts of the model representing a principal's state have a behavioral interpretation.

6.1 More on NSTIC

We discussed the White House's "National Strategy for Trusted Identities in Cyberspace" (NSTIC) in Chapter 1. I claimed that NSTIC's mechanisms for online identification and authentication are riskier than those they are meant to replace, because they require more trust on the part of users. Here, I will give an example of this additional trust. (While I was working on this thesis, Ross Anderson presented a paper ([44]) about similar problems with federated authentication in general, in which he mentions NSTIC. I highly recommend it to anyone interested in the practical difficulties with requiring a lot of trust from users.)

There are multiple kinds of trust required by the White House's mechanisms that are not re-

quired by traditional mechanisms, but one of the more important ones occurs in an example provided by the White House itself in its discussion of the privacy-relevant aspects of its mechanisms:

> The offline world has structural barriers that preserve individual privacy by limiting information collection, use, and disclosure to a specific context. For example, consider a driver's license: an individual can use a driver's license to open a bank account, board an airplane, or view an age-restricted movie at the cinema, but the Department of Motor Vehicles does not know every place that accepts driver's licenses as identification. It is also difficult for the bank, the airport, and the movie theater to collaborate and link the transactions together. At the same time, there are aspects of these offline transactions that are not privacy-protective. The movie theater attendant who checks an individual's driver's license needs to know only that the individual is over age 17. But looking at the driver's license reveals extraneous information, such as the individual's address and full date of birth. [1, p. 11]

As they say, the White House's mechanisms ideally "should preserve the positive privacy benefits of offline transactions while mitigating some of the negative privacy aspects" (ibid.), and of course they should do the same for the traditional online mechanisms that they replace. However, while the White House's mechanisms *might* be able to limit extraneous information given to service providers without additional trust requirements (but probably not, although it depends on the details of the mechanism), they will not be able to prevent identity providers like the DMV from retaining information about the services that users use. They replace a physical guarantee of privacy with a requirement that the user trusts the identity provider to protect the user's privacy by behaving well — specifically, by abiding by the "Fair Information Practice Principles" (cf. ibid., p. 45).

Something similar will happen when an NSTIC mechanism replaces digital mechanisms like password-based authentication. Right now, my online bank account is protected with a username-password pair, and my bank and I know that, if the security mechanisms work, only we can access that account. But in the NSTIC vision, my bank will depend on an assertion from an identity provider to authenticate me, and therefore to be satisfied that only I or the bank will access my account, we must trust the identity provider not to accidentally or deliberately misidentify someone else as me. (An interesting question is, if someone gets into my account somehow and steals money from it, how liable are the bank and the identity provider? For more

on this theme, cf. [44].)

6.2 Future Work

The most pressing future work is to address the issues with the belief model, discussed at the end of Chapter 5. We may need to abandon the principle that all parts of a local state have behavioral interpretations, and represent beliefs directly in local states. This then raises the question of how to represent beliefs. BAN Logic and Rangan's method include formulas from the formal languages in the models, and this, at least on the surface, seems inappropriate. Whatever approach is taken, the result should be a model with clear criteria for determining whether it applies to a given real-world situation — in short, it should be meaningful.

In Chapter 4, we did not go far in understanding the differences between the two meanings of 'call'; we also did not consider whether there might be other types of names relevant to security protocols. An interesting future project would be to list all the types of names that security protocols use, and also to investigate the usefulness of the concepts in the philosophical literature on naming such as [25], [28], and [29] with regard to understanding the types of names used in security protocols.

Finally, the ultimate goal would be to present the required assumptions of a representative collection of widely used security protocols, including multiple protocols for a given purpose (e.g., authentication). This would be very useful for understanding all the questionable assumptions we are making without even knowing it whenever we use our computers.

THIS PAGE INTENTIONALLY LEFT BLANK

REFERENCES

[1] White House, "National strategy for trusted identities in cyberspace," Apr. 2011. [Online]. Available: http://www.nist.gov/nstic/

[2] E. Maler and D. Reed, "The Venn of identity: Options and issues in federated identity management," *IEEE Security and Privacy*, vol. 6, no. 2, pp. 16–23, Mar.–Apr. 2008.

[3] R. Dhamija and L. Dusseault, "The seven flaws of identity management: Usability and security challenges," *IEEE Security and Privacy*, vol. 6, no. 2, pp. 24–29, Mar.–Apr. 2008.

[4] D. Davidson, "Quotation," *Theory and Decision*, vol. 11, no. 1, pp. 27–40, Mar. 1979.

[5] W. V. Quine, *Mathematical Logic, Revised Edition*. Harvard University Press, 1981.

[6] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi, *Reasoning About Knowledge*. MIT Press, 1995.

[7] S. A. Kripke, "Semantical analysis of modal logic I: Normal modal propositional calculi," *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, vol. 9, no. 5–6, pp. 67–96, 1963.

[8] M. Huth and M. Ryan, *Logic in Computer Science*, 2nd ed. Cambridge University Press, 2004.

[9] J. LaPorte, "Rigid designators," in *The Stanford Encyclopedia of Philosophy*, Summer 2011 ed., E. N. Zalta, Ed., 2011, forthcoming. [Online]. Available: http://plato.stanford.edu/archives/sum2011/entries/rigid-designators/

[10] A. J. Grove, "Naming and identity in epistemic logic part II: A first-order logic for naming," *Artificial Intelligence*, vol. 74, no. 2, pp. 311–350, Apr. 1995.

[11] R. C. Barcan, "A functional calculus of first order based on strict implication," *The Journal of Symbolic Logic*, vol. 11, no. 1, pp. 1–16, Mar. 1946.

[12] P. V. Rangan, "An axiomatic basis of trust in distributed systems," in *Proceedings of the 1988 IEEE Symposium on Security and Privacy*. IEEE Computer Society, Apr. 1988, pp. 204–211.

[13] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *Proceedings of the Royal Society of London A*, vol. 426, no. 1871, pp. 233–271, 8 Dec. 1989, a preliminary version appeared in February 1989 as Research Report 39, DEC Systems Research Center, Palo Alto, CA.

[14] L. E. Moser, "A logic of knowledge and belief for reasoning about computer security," in *Proceedings of Computer Security Foundations Workshop II*. IEEE Computer Society, Jun. 1989, pp. 57–63.

[15] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," in *Proceedings of the Twelfth ACM Symposium on Operating Systems Principles (SOSP '89)*. ACM, Nov. 1989, pp. 1–13, also published as [30].

[16] ——, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18–36, Feb. 1990.

[17] L. Gong, R. Needham, and R. Yahalom, "Reasoning about belief in cryptographic protocols," in *Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy*. IEEE Computer Society, May 1990, pp. 234–248.

[18] R. Kailar and V. D. Gligor, "On belief evolution in authentication protocols," in *Proceedings of Computer Security Foundations Workshop IV*. IEEE Computer Society, Jun. 1991, pp. 103–116.

[19] M. Abadi and M. R. Tuttle, "A semantics for a logic of authentication (extended abstract)," in *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing (PODC '91)*. ACM, Aug. 1991, pp. 201–216.

[20] R. Yahalom, B. Klein, and T. Beth, "Trust relationships in secure systems - a distributed authentication perspective," in *Proceedings of the 1993 IEEE Symposium on Research in Security and Privacy*. IEEE Computer Society, May 1993, pp. 150–164.

[21] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," in *Proceedings of the 1996 IEEE Symposium on Security and Privacy*. IEEE Computer Society, May 1996, pp. 164–173.

[22] N. Li, J. C. Mitchell, and W. H. Winsborough, "Design of a role-based trust-management framework," in *Proceedings of the 2002 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2002, pp. 114–130.

[23] J. D. Guttman, F. J. Thayer Fábrega, J. A. Carlson, J. C. Herzog, J. D. Ramsdell, and B. T. Sniffen, "Trust management in strand spaces: A rely-guarantee method," in *Programming Languages and Systems*, ser. Lecture Notes in Computer Science, D. Schmidt, Ed. Springer Berlin / Heidelberg, 2004, vol. 2986, pp. 325–339.

[24] A. W. Appel and E. W. Felten, "Proof-carrying authentication," in *Proceedings of the 6th ACM Conference on Computer and Communications Security (CCS '99)*. ACM, 1999, pp. 52–62.

[25] G. Frege, "Über sinn und bedeutung," *Zeitschrift für Philosophie und philosophische Kritik*, vol. 100, pp. 25–50, 1892, English translation by Max Black as "Sense and Reference", *The Philosophical Review*, vol. 57, no. 3, pp. 209–230, 1948.

[26] W. V. Quine, "Notes on existence and necessity," *The Journal of Philosophy*, vol. 40, no. 5, pp. 113–127, 4 Mar. 1943.

[27] ——, "The problem of interpreting modal logic," *The Journal of Symbolic Logic*, vol. 12, no. 2, pp. 43–48, Jun. 1947.

[28] ——, "Quantifiers and propositional attitudes," *The Journal of Philosophy*, vol. 53, no. 5, pp. 177–187, 1 Mar. 1956.

[29] D. Kaplan, "Quantifying in," *Synthese*, vol. 19, no. 1–2, pp. 178–214, Dec. 1968.

[30] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *SIGOPS Operating Systems Review*, vol. 23, no. 5, pp. 1–13, Dec. 1989, also published as [15].

[31] E. Snekkenes, "Exploring the BAN approach to protocol analysis," in *Proceedings of the 1991 IEEE Symposium on Research in Security and Privacy*. IEEE Computer Society, May 1991, pp. 171–181.

[32] P. Syverson, "The use of logic in the analysis of cryptographic protocols," in *Proceedings of the 1991 IEEE Symposium on Research in Security and Privacy*. IEEE Computer Society, May 1991, pp. 156–170.

[33] J. Steiner, C. Neuman, and J. I. Schiller, "Kerberos: An authentication service for open network systems," in *USENIX Winter 1988 Technical Conference*. USENIX Association, 1988.

[34] C. M. Ellison, "SPKI requirements," RFC 2692 (Experimental), Sep. 1999. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2692.txt

[35] C. M. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen, "SPKI certificate theory," RFC 2693 (Experimental), Sep. 1999. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2693.txt

[36] International Telecommunication Union, "Information technology — open systems interconnection — the directory: Authentication framework," ITU-T Recommendation X.509, Aug. 1997. [Online]. Available: http://www.itu.int/rec/T-REC-X.509

[37] A. J. Grove and J. Y. Halpern, "Naming and identity in epistemic logics part I: The propositional case," *Journal of Logic and Computation*, vol. 3, no. 4, pp. 345–378, Aug. 1993.

[38] D. Lewis, "Attitudes *de dicto* and *de se*," *The Philosophical Review*, vol. 88, no. 4, pp. 513–543, Oct. 1979.

[39] Y. Lesperance, "A formal account of self knowledge and action," in *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, vol. 2. The International Joint Conferences on Artificial Intelligence, Inc., 1989, pp. 868–874.

[40] F. J. Thayer Fábrega, J. C. Herzog, and J. D. Guttman, "Strand spaces: Proving security protocols correct," *Journal of Computer Security*, vol. 7, no. 2–3, pp. 191–230, 1999.

[41] OpenID Foundation, "OpenID Authentication 2.0 - final," Dec. 2007, the specifications for the other parts of OpenID are at http://openid.net/developers/specs/. [Online]. Available: http://openid.net/specs/openid-authentication-2_0.html

[42] D. Hardt, J. Bufu, and J. Hoyt, "OpenID Attribute Exchange 1.0 - final," Dec. 2007. [Online]. Available: http://openid.net/specs/openid-attribute-exchange-1_0.html

[43] E. Rescorla, "Diffie-Hellman key agreement method," RFC 2631, Jun. 1999. [Online]. Available: http://www.rfc-editor.org/rfc/rfc2631.txt

[44] R. Anderson, "Can we fix the security economics of federated authentication?" 28 Mar. 2011, presented at the Nineteenth International Workshop on Security Protocols in Cambridge, England.

Referenced Authors

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Fort Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Prof. Bret Michael
   Naval Postgraduate School
   Monterey, California

4. Prof. George Dinolt
   Naval Postgraduate School
   Monterey, California

5. Dr. Paul Syverson
   Naval Research Laboratory
   Washington, District of Columbia

6. Prof. Joshua Guttman
   Worcester Polytechnic Institute
   Worcester, Massachusetts

7. Dr. Ed Zieglar
   National Security Agency
   Fort Meade, Maryland

8. Prof. Duminda Wijesekera
   George Mason University
   Fairfax, Virginia